© 2014 ZIH Corp.  The copyrights in this manual and the software and/or firmware in the printer described therein are owned by ZIH Corp. Unauthorized reproduction of this manual or the software and/or firmware in the printer may result in imprisonment of up to one year and fines of up to $10,000 (17 U.S.C.506). Copyright violators may be subject to civil liability.

This product may contain ZPL®, ZPL II®, and ZebraLink™ programs; Element Energy Equalizer® Circuit; E³®; and Monotype Imaging fonts. Software © ZIH Corp. All rights reserved worldwide.

ZebraLink and all product names and numbers are trademarks, and Zebra, the Zebra logo, ZPL, ZPL II, Element Energy Equalizer Circuit, and E³ Circuit are registered trademarks of ZIH Corp. All rights reserved worldwide.

All other brand names, product names, or trademarks belong to their respective holders. For additional trademark information, please see “Trademarks” on the product CD.

Proprietary Statement  This manual contains proprietary information of Zebra Technologies Corporation and its subsidiaries (“Zebra Technologies”). It is intended solely for the information and use of parties operating and maintaining the equipment described herein. Such proprietary information may not be used, reproduced, or disclosed to any other parties for any other purpose without the express, written permission of Zebra Technologies Corporation.

Product Improvements  Continuous improvement of products is a policy of Zebra Technologies Corporation. All specifications and designs are subject to change without notice.

Liability Disclaimer  Zebra Technologies Corporation takes steps to ensure that its published Engineering specifications and manuals are correct; however, errors do occur. Zebra Technologies Corporation reserves the right to correct any such errors and disclaims liability resulting therefrom.

Limitation of Liability  In no event shall Zebra Technologies Corporation or anyone else involved in the creation, production, or delivery of the accompanying product (including hardware and software) be liable for any damages whatsoever (including, without limitation, consequential damages including loss of business profits, business interruption, or loss of business information) arising out of the use of, the results of use of, or inability to use such product, even if Zebra Technologies Corporation has been advised of the possibility of such damages. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.
Contacts

Technical Support

Technical Support is available via Internet 24 hours per day, 365 days per year at www.zebra.com. You can also email or call us using the following contact information.

The Americas - kiosksupport@zebra.com
Europe, Middle East, and Africa (EMEA) - tseurope@zebra.com
China - tschina@zebra.com
Asian Pacific (except China) and India - tsasiapacific@zebra.com

Zebra Technologies Corporation
Zebra Technologies Corporation
475 Half Day Road, Suite 500
Lincolnshire, IL 60069 USA
T: +1 847 634 6700
Toll-free +1 866 230 9494
F: +1 847 913 8766

Zebra Technologies Europe Limited
Dukes Meadow
Millboard Road
Bourne End
Buckinghamshire, SL8 5XF, UK
T: +44 (0)1628 556000
F: +44 (0)1628 556001

Zebra Technologies Asia Pacific, LLC
120 Robinson Road
#06-01 Parakou Building
Singapore 068913
T: +65 6858 0722
F: +65 6885 0838

Additional Links

<table>
<thead>
<tr>
<th>To find...</th>
<th>go to...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support &amp; Downloads</td>
<td><a href="http://www.zebra.com/support">http://www.zebra.com/support</a></td>
</tr>
<tr>
<td>Customer Service and General Inquires</td>
<td><a href="http://www.zebra.com/howtobuy">http://www.zebra.com/howtobuy</a></td>
</tr>
<tr>
<td>Knowledge Base</td>
<td><a href="http://km.zebra.com">http://km.zebra.com</a></td>
</tr>
<tr>
<td>Repair Order (RO) Request and Repair Services</td>
<td><a href="http://www.zebra.com/repair">http://www.zebra.com/repair</a></td>
</tr>
<tr>
<td>Technical Training</td>
<td><a href="http://www.zebra.com/training">http://www.zebra.com/training</a></td>
</tr>
</tbody>
</table>
Contents

Windows Statuses .......................................................... 53
  Windows Compatible Status ........................................... 53
  Statuses Defined in winspool.h .................................... 53
  Windows Incompatible Status ....................................... 55

GetPrinterData Key Values ............................................. 57
  GetPrinterData Key Values ......................................... 57

Status Monitoring & Programming Examples ......................... 59
  Status Monitoring ................................................. 59
  Implementation in Calling Application ......................... 60
  Implementation in Monitor Thread for OS Prior to Windows 7 .... 61
  Implementation in Monitor Thread for OS Windows 7 and Above .... 63
    Status function called from within the monitoring thread: .... 63
    Status Monitoring Thread ..................................... 66
  Status Implementation with C# .................................. 67
    WMI Script to get Basic Status ................................ 72

Print Forms .................................................................. 75
  Setup Print Forms in Windows XP and Vista ..................... 75
    Viewing and Creating Print Forms ............................... 76
  Setup Print Forms in Windows 7 .................................. 77
  Additional References .............................................. 79

Index ........................................................................... 81
Introduction

About this Manual

This manual is updated from time to time when printer functions and features are added or amended. You can find the latest edition on our website at www.zebra.com. If you require functions not found in this manual edition, please contact Technical Support for your region or the Zebra partner from which you purchased the printer.
Zebra Kiosk Printer Driver Installation

Contents
Zebra Kiosk Printer Driver Installation ........................................ 9
Zebra Kiosk Printer Driver Functionality ...................................... 22
Zebra Kiosk Printer Driver Properties ....................................... 23

Zebra Kiosk Printer Driver Installation
The Zebra Kiosk Printer Driver Installer installs the driver files on your hard disk and pre-installs the drivers for the KR203, TTP 2000 series, TTP 2100 series, TTP 7030, and TTP 8000 series printers. This enables you to easily setup your Zebra Kiosk printer!

The Zebra Kiosk Printer Driver Installer procedure requires the following steps:
• Step 1: Uninstall the Old Kiosk Drivers (if applicable)
• Step 2: Pre-install the New Kiosk Drivers Before Connecting Printer

Step 1: Uninstall the Old Kiosk Drivers (if applicable)
If you have any old Zebra Kiosk printer drivers installed on your system, you need to uninstall those drivers prior to installing the new drivers. If you do not have any old drivers installed, go to Step 2: Pre-install the New Kiosk Drivers Before Connecting Printer.

If you have old drivers installed, follow the appropriate procedure to uninstall those drivers.
• Windows XP Uninstall
• Windows 7 Uninstall
Windows XP Uninstall

If you are running Windows XP, you can use the Windows Driver Uninstaller to remove the old drivers prior to installing the new drivers or you can manually uninstall the old drivers.

Using the Windows Driver Uninstaller

The zebra\kiosk\WindowsDriver\TTP folder contains a shortcut to the Windows Driver Uninstaller.

1. Double-click windows-driver-uninstall.exe.

   The Driver Un-Install Program dialog appears.

2. In the Printer list, select the printer that you want to uninstall.

3. In the Driver list, select the driver that you want to uninstall.

4. In the Paper forms list, select all of the paper forms for that driver.

5. Click un-install.

Note • You can also download the Windows Driver Uninstaller from www.zebra.com/support. Select the printer model from the Printer Support list, click the Software Utilities tab, and click Download next to Windows Driver Uninstaller.
The following dialog appears asking if you want to restart your computer.

![Restart Dialog]

6. Click **Yes** to restart your computer. This is required prior to installing the new driver.

**Using the Manual Uninstall Procedure**

1. Click **Start**, and then click **Printers and Faxes**.

2. Right-click the printer that you want to uninstall, and click **Delete**.

![Printers and Faxes]

3. On the **File** menu, click **Server Properties**.

![Server Properties]

The **Server Properties** dialog appears.
4. Click the **Forms** tab.

5. In the **Forms on** list, scroll down to locate a form that is specific to the Kiosk printer and is not a system form (for example, A4 1/3). The **Delete** button becomes available indicating that it is not a system form.

6. Click the form, and then click **Delete**. Repeat this step for each form in the list that is not a system form.

7. Click the **Drivers** tab.

8. Click the driver that you want to uninstall, and then click **Remove**.

10. In Windows Explorer, open the C:\Windows\inf folder.

11. Click the Search button.

12. In the All or part of the file name box, type OEM*.inf.

13. In the A word or phrase in the file box, type Swecoin, and click Search.

   The Search Results show the oem*.inf file(s) that need to be deleted.

14. In Windows Explorer, open the C:\Windows\inf folder and select the resulting oem*.inf file(s) and the matching oem*.PNF file(s), then right-click your selection and click Delete.

   Note • The PNF files are precompiled versions of the INF files and must also be deleted.
Windows 7 Uninstall

If you are running Windows 7 Professional or Ultimate (32-bit or 64-bit), use the Print Management dialog to uninstall the old drivers.

**Note** • You must be signed in as an Administrator to use Print Management.

1. Click Start, and in the search box type printmanagement.msc, and then press Enter. The Print Management dialog appears.

2. In the left pane, click All Printers to display the printer list.

3. In the printer list, select each of the Zebra printers, right-click your selection, and click Delete.
   The following message appears asking you to confirm the deletion.

4. Click Yes to confirm the deletion and return to the Print Management dialog.
5. In the left pane, click **All Drivers** to display the driver list.

6. Right-click the Zebra driver that you want to uninstall, and click **Remove Driver Package**.

   This removes all of the Zebra drivers in this package (i.e., KR203, TTP 2000 series, TTP 2100 series, TTP 7030, and TTP 8000 series).

   **Note** • This only works if all of the printers have been uninstalled first. If you have not uninstalled the printers, you will receive a message indicating that the driver cannot be deleted. Uninstall the printers as described above and then repeat this step.

   The following **Print Management** message appears asking you to confirm the deletion.

7. Click **Delete**.
The following **Print Management** message appears indicating that the driver package was deleted, and shows which drivers were removed.

![Print Management dialog](image)

8. Click **OK** to complete the uninstall.

9. Close the **Print Management** dialog.

**Step 2: Pre-install the New Kiosk Drivers Before Connecting Printer**

After **Step 1: Uninstall the Old Kiosk Drivers (if applicable)** is complete, use the **Zebra Kiosk Printer Driver 1.3.510.XX Installer** to pre-install the new drivers.

To download the **Kiosk Printer Driver version 1.3.510.XX from the Zebra website**

2. Select your printer from the **Printer Support** list.
3. Click the **Drivers** tab.
4. Click **Download** to download the **Kiosk Printer Driver** to your computer. The **Zebra Kiosk Printer Driver Installer** icon appears on your Desktop.
To run the Kiosk Printer Driver version 1.3.510.XX installation

1. On the Desktop, double-click the Zebra Kiosk Printer Driver Installer icon to start the InstallAware Wizard.

After the contents of the setup package are verified the Welcome screen appears.

2. Click Next.

Notes • If you have previously installed the driver package a different Welcome dialog appears. Select Repair Installation, and then click Next.
You will not see the End User License Agreement or the Important Information dialogs shown on the following page.
The End User License Agreement appears.

3. Select the I accept the terms of the license agreement check box, and click Next.
The following Important Information appears. This information contains the Zebra Kiosk Printer Windows driver release notes.

4. Read the information, and then click Next.

**Note** • The Readme file also contains the Zebra Kiosk Printer Windows driver release notes. To open the Readme file, click Start > All Programs > Zebra Technologies > Zebra Kiosk Printer Driver 1.3.510.XX > Readme.
The application starts copying the driver files to the driver directory.

![Image](image1.png)

**Note** • If you are running Windows XP, a **Files Needed** message may appear asking you to locate a particular .GPD file. Locate the file and click **OK** to continue the installation.

After the files are copied, the following dialog appears indicating that the installation is completing. The copied files are located in **C:\Zebral\kiosk\WindowsDriver\TTP**.

![Image](image2.png)

5. Click **Finish** to complete the driver package installation.
Now all of the drivers for the KR203, TTP 2000 series, TTP 2100 series, TTP 7030, and TTP 8000 series printers are pre-installed and ready to use with your printer.
6. Plug your printer into the USB or serial port.

7. Power on the printer.

**Note** • If the printer drivers do not successfully pre-install, or if you have a non Plug and Play printer with parallel or serial ports, you will need to use the **Windows Add Printer Wizard** to install your printer.

**Note**
If you are installing the printer drivers on a **Windows Embedded** operating system (e.g., Windows Embedded XPe and Windows Embedded POSReady 7), use the following Microsoft links for instructions on how to create an image and it’s driver components.

- The following link opens the Installation Guide.
  

- The TTP.inf file should be installed into the Component Manager. The link below provides instructions on how to componentize a third-party driver. This is where the TTP.inf file should be imported.
  

- The following link provides instructions on how to create the run-time image.
  

- When the XPe image is fully booted up on the client box and the printer wizard appears, you are prompted to install the following .DLLs.
  
  - TTPRES.DLL - points the printer install wizard to the path where the driver install folder is put
  
  - UNIDRIV.DLL - located at `C:\Windows\System32\spool\drivers\W32X86\3`
  
  - UNIRES.DLL - located at `C:\Windows\System32\spool\drivers\W32X86\3`
The printer appears in the **Printers and Faxes** area of the Device and Printers dialog and is now ready for use.
Zebra Kiosk Printer Driver Functionality

The Zebra Kiosk Printer Windows driver is based on the Microsoft Unidriver architecture for raster based printers. Zebra provides two OEM libraries (UI and Rendering) to enable specific printer functionalities within the driver. In addition to the standard Microsoft driver, Zebra provides a bi-directional interface through a Language Monitor DLL.

Due to the function compatibility of the different printer families (KR203, TTP 2000, TTP 2100, TTP 7030, and TTP 8000) the drivers share many functions in the UI and Rendering DLL as well as the Language Monitor. All of the OEM features are described below.

**Note** • If you are uploading firmware via the Zebra Toolbox program you need to ensure that the Enable bidirectional support check box is cleared and the spooler is restarted. If you do not restart the spooler the change will not take effect!
Zebra Kiosk Printer Driver Properties

The KR203, TTP 2000 series, TTP 2100 series, TTP 7030, and TTP 8000 series have the same basic Properties dialog. The Properties dialog tabs are described in the following sections. Windows 7 dialogs are used for these descriptions.

- General
- Sharing
- Ports
- Advanced
- Color Management
- Security
- Device Settings
- Tools
- Printer Information
- Import/Export settings
- About

Note •
In Windows 7, you MUST use the Print Management dialog to make changes to the Properties dialog Device Settings tab. This is because you must have Administrative rights. To open the Print Management dialog, click Start, and in the search box type printmanagement.msc, and then press Enter.
The General tab shows the name, location, and features of the printer. It also enables you to set preferences and print a test page.

- On the General tab, click Preferences to open your printer’s Printing Preferences dialog.
Printing Preferences

The Printing Preferences dialog has two tabs: Layout and Paper/Quality.

- On the Layout tab, you can select the orientation and the page order.

- On the Paper/Quality tab, you can select the paper source and color.
• Click **Advanced** to open the **Advanced Options** dialog where you can set your paper size and count, graphic scaling, and document options as described in the following sections. The options that are available depend on your printer.

### TTP 8200

![TTP 8200 Advanced Options](image)

### TTP 2030

![TTP 2030 Advanced Options](image)

### Paper/Output

#### Paper Size

The printer series have different **Paper Size** choices.

<table>
<thead>
<tr>
<th>KR203</th>
<th>TTP 2000</th>
<th>TTP 2100</th>
<th>TTP 7030</th>
<th>TTP 8200</th>
<th>TTP 8300</th>
</tr>
</thead>
<tbody>
<tr>
<td>58 mm x 400 mm Roll Paper</td>
<td>60 mm Roll Paper</td>
<td>30.5 mm Ticket</td>
<td>80 mm Roll Paper</td>
<td>82.5 mm Roll Paper</td>
<td>112 mm Roll Paper</td>
</tr>
<tr>
<td>60 mm x 400 mm Roll Paper</td>
<td>60 mm Roll Paper</td>
<td>54 mm Ticket</td>
<td>60 mm Ticket</td>
<td>80 mm Roll Paper</td>
<td>82.5 mm Ticket</td>
</tr>
<tr>
<td>80 mm x 150 mm Roll Paper</td>
<td>60 mm Ticket</td>
<td>66 mm Ticket</td>
<td>80 mm Ticket</td>
<td>82.5 mm Ticket</td>
<td>123 mm Roll Paper</td>
</tr>
<tr>
<td>80 mm x 250 mm Roll Paper</td>
<td>82.5 mm Ticket</td>
<td>82.5 mm Ticket</td>
<td>216 mm Roll Paper</td>
<td>230 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
</tr>
<tr>
<td>80 mm x 400 mm Roll Paper</td>
<td>123 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
</tr>
<tr>
<td>82.5 mm x 127 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
</tr>
<tr>
<td>82.5 mm x 254 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
</tr>
<tr>
<td>82.5 mm x 400 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
<td>216 mm Roll Paper</td>
</tr>
</tbody>
</table>

#### Copy Count

The **Copy Count** option enables you to specify the number of copies to print.
Graphic

Scaling

The Scaling option enables you to change the size of your printable area. When you scale down, you can print larger pages on smaller paper. When you scale up, you can print smaller pages on larger paper.

Document Options

Advanced Printing Features

This is a Microsoft Unidriver setting and should always be set to Enabled.

Page per Sheet Layout

This is a Microsoft Unidriver setting and should always be set to Right then Down.

Color Printing Mode

This OEM setting allows you to select one of two currently available dithering modes.

- OEM Color Quality mode is the default and does a dithering similar to a Riemersma dither algorithm with a gray scaling effect.
- B/W Quality mode uses a Threshold dithering algorithm that only displays black and white areas.

Printer Features

Vertical Mount

The Vertical Mount option enables you to select which way you want to mount the printer. The default is Off. Select On if you want to mount your printer in the vertical position.

Low Temperature Compensation

If the printer is located in a cold area, set the Low Temperature Compensation option to On. The default is Off.
Sharing

The **Sharing** tab enables you to share your printer with other computers on a Network. The **Sharing** tab also enables you to install additional drivers for users that are running different versions of Windows.
Ports

The **Ports** tab shows to which port the printer is connected. The **Ports** tab also enables you to add, delete, and configure ports. The **Ports** tab is the same for all printer series.

- Select the **Enable bidirectional support** check box to control the functionality of the Language Monitor.
- Clear the **Enable printer pooling** check box. This feature is not used for Kiosk printing.

**Note** • If you are uploading firmware via the Zebra Toolbox program you need to ensure that the **Enable bidirectional support** check box is cleared and the spooler is restarted. If you do not restart the spooler the change will not take effect!
Advanced

The Advanced tab enables you to specify when the printer is available, select the printer driver, and set spooling options. In addition you can open Printing Defaults, Print Processor, and Separator Page dialogs from this tab.
Color Management

The Color Management tab settings are specific to Microsoft Universal Printer Drivers (UniDrv). The default settings should be used.
Security

The Security tab enables you to set the access control of specific system users for your printer. In some cases where you need to lock down your user account (e.g., in Kiosk applications) you need to grant the Kiosk user full administrator access to the printer. Typically a “normal” user has only Print rights but in order to get status from the printer the user also needs Manage Printer and Manage Documents permissions.
Device Settings

The Device Settings tab enables you to set various printer, document, presenter, and driver settings. These settings are similar between printer series, although some differences do apply.

The Minimum, Maximum, and Default Settings table shows minimum, maximum, and default settings for each printer family.

The following sections describes each of the Device Settings.

- Form To Tray Assignment
- Printer Settings
- Document Settings
- Presenter Settings
- Driver Settings
## Minimum, Maximum, and Default Settings

The following table shows minimum and maximum (where applicable), and default printer settings for each printer family. The default value is shown in **bold** font.

<table>
<thead>
<tr>
<th>Printer Setting</th>
<th>KR203</th>
<th>TTP 2000</th>
<th>TTP 2100</th>
<th>TTP 7030</th>
<th>TTP 8200</th>
<th>TTP 8300</th>
</tr>
</thead>
<tbody>
<tr>
<td>Form To Tray Assignment</td>
<td>Default: 80 mm x 400 mm Roll Paper</td>
<td>Default: 80 mm Roll Paper</td>
<td>Default: 80 mm Roll Paper</td>
<td>Default: 112 mm Roll Paper</td>
<td>Default: Letter</td>
<td>Default: Letter</td>
</tr>
<tr>
<td>(see Form To Tray Assignment for options)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Media Tracking</td>
<td>Default: Variable length</td>
<td>Default: Variable length</td>
<td>Default: Continuous</td>
<td>Default: Variable length</td>
<td>Default: Continuous</td>
<td>Default: Continuous</td>
</tr>
<tr>
<td>Bottom margin</td>
<td>Default: 0 Min: 0 Max: 9</td>
<td>Default: 0 Min: 0 Max: 9</td>
<td>Default: 0 Min: 0 Max: 9</td>
<td>Default: 0 Min: 0 Max: 19</td>
<td>Default: 0 Min: 0 Max: 19</td>
<td>Default: 0 Min: 0 Max: 19</td>
</tr>
<tr>
<td>Cutter mode</td>
<td>Default: Cut Per Page</td>
<td>Default: Cut Per Page</td>
<td>Default: Cut Per Page</td>
<td>Default: Cut Per Page</td>
<td>Default: Cut Per Page</td>
<td>Default: Cut Per Page</td>
</tr>
<tr>
<td>Partial Cut Width</td>
<td>Default: 0 Min: 0, 10 Max: 60</td>
<td>Default: 0 Min: 0, 10 Max: 40</td>
<td>Default: 0 Min: 0, 10 Max: 40</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Presenter loop length</td>
<td>Default: 400 Min: 0 Max: 600</td>
<td>Default: 480 Min: 96 Max: 8160</td>
<td>N/A</td>
<td>N/A</td>
<td>Default: 320 Min: 96 Max: 8160</td>
<td>Default: 320 Min: 96 Max: 8160</td>
</tr>
<tr>
<td>Present Length Addition</td>
<td>Default: 0 Min: 0 Max: 255</td>
<td>Default: 0 Min: 0 Max: 255</td>
<td>Default: 0 Min: 0 Max: 255</td>
<td>Default: 0 Min: 0 Max: 255</td>
<td>Default: 0 Min: 0 Max: 255</td>
<td>Default: 0 Min: 0 Max: 255</td>
</tr>
<tr>
<td>Presenter mode</td>
<td>N/A</td>
<td>Default: Eject</td>
<td>N/A</td>
<td>Default: Eject</td>
<td>Default: Eject</td>
<td>Default: Eject</td>
</tr>
</tbody>
</table>
The **Form To Tray Assignment** setting shows the currently selected paper form. You can select from a variety of paper forms and custom forms generated in **Server Properties** dialog (see **Print Forms**). Set this setting the same as set in the **Printing Preferences** dialog.

<table>
<thead>
<tr>
<th>Printer Setting</th>
<th>KR203</th>
<th>TTP 2000</th>
<th>TTP 2100</th>
<th>TTP 7030</th>
<th>TTP 8200</th>
<th>TTP 8300</th>
</tr>
</thead>
<tbody>
<tr>
<td>Presenter timeout</td>
<td>Default: 0 Min: 0 Max: 300</td>
<td>Default: 0 Min: 0 Max: 300</td>
<td>N/A</td>
<td>Default: 0 Min: 0 Max: 300</td>
<td>Default: 0 Min: 0 Max: 300</td>
<td>Default: 0 Min: 0 Max: 300</td>
</tr>
<tr>
<td>Clear presenter</td>
<td>Default: No</td>
<td>Default: No</td>
<td>Default: No</td>
<td>Default: No</td>
<td>Default: No</td>
<td>Default: No</td>
</tr>
<tr>
<td>Page hold</td>
<td>Default: No</td>
<td>Default: No</td>
<td>Default: No</td>
<td>Default: No</td>
<td>Default: No</td>
<td>Default: No</td>
</tr>
<tr>
<td>Image Adjustment</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Default: No</td>
<td>Default: No</td>
</tr>
<tr>
<td>Contrast</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Default: 0 Min: -100 Max: 100</td>
<td>Default: 0 Min: -100 Max: 100</td>
</tr>
<tr>
<td>Brightness</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>Default: 0 Min: -100 Max: 100</td>
<td>Default: 0 Min: -100 Max: 100</td>
</tr>
</tbody>
</table>

The following sections show the available forms for each printer family.
**KR203**

The following forms are available for the KR203. The pre-defined forms have a length of 40.64 cm or 16 inch.

![KR203 Device Settings Diagram](image)

**TTP 2000**

The following forms are available for the TTP 2000 series. The pre-defined forms have a length of 40.64 cm or 16 inch.

![TTP 2000 Device Settings Diagram](image)
TTP 2100

The following forms are available for the TTP 2100 series. The pre-defined forms have a length of 15 cm or 5.91 inch or the specific form length of the bag tag or ticket.

TTP 7030

The following forms are available for the TTP 7030. The pre-defined forms have a length of 40.6 cm or 16 inch.


**TTP 8200**

The following forms are available for the TTP 8200 series. The pre-defined forms have a length of the specific form length.


**TTP 8300**

The following forms are available for the TTP 8300 series. The pre-defined forms have a length of the specific form length.
Printer Settings

The Printer Settings enable you to set the Darkness and the Max print speed (mm/s).

- Darkness
  The Darkness setting affects Printer Parameter 7 each time a print job is issued. The minimum, maximum, and default values are shown in the Minimum, Maximum, and Default Settings table.

- Print speed
  The Print speed affects Printer Parameter 8 each time a print is issued. The minimum, maximum, and default values are shown in the Minimum, Maximum, and Default Settings table.

Note • See the Technical Manual for your printer for more information on these settings.
**Document Settings**

The **Document Settings** enable you to set the **Media Tracking**, **Top margin**, and **Bottom margin** as described in the following sections.

![Zebra TTP 8200 (Copy 1) Properties](image)

- **Media Tracking**
  
  The **Media Tracking** setting determines how the media is delimited. The possible values are Continuous, Variable length, and Mark sensing.
  
  - **Continuous** — always prints a whole page
  - **Variable length** — cuts white space at the end
  - **Mark sensing** — syncs with black marks

  The default values are shown in the Minimum, Maximum, and Default Settings table.

- **Top margin**
  
  The **Top margin** setting affects the physical distance between the top of the paper and the cutter. Due to the mechanical design, the printer will always have a top margin depending on the printer family. This distance between the cutter and the print head can be reduced by reversing the paper. The value entered in this setting determines the amount the printer has to reverse paper (see the Technical Manual description of the ESC j command). The minimum, maximum, and default values are shown in the Minimum, Maximum, and Default Settings table.

  **Note** - The physical distance for a TTP 2000 is 9 mm, for a TTP 2100 is 9 mm, for a TTP 7030 is 14 mm and for a TTP 8200 and TTP 8300 is 19 mm.
• **Bottom margin**

The **Bottom margin** setting affects the physical distance between the bottom of the paper and the cutter. This setting is an addition to the actual page length in Variable mode and restricts the printable page in Continuous mode and Black Mark mode.

The minimum, maximum, and default values are shown in the **Minimum, Maximum, and Default Settings** table.

**Presenter Settings**

The **Presenter Settings** enable you to set various presenter settings depending upon your printer family as described in the following sections.

• **Cutter mode**

The **Cutter mode** setting is a driver only setting and does not affect any Printer Parameters. The possible values are **Cut every page**,** Cut at the document end**, and **No Cut** (not advisable).

• **Cut every page** — driver issues a cut command after every page of a document
• **Cut at the document end** — driver issues a cut only at the end of a document
• **No Cut** — driver does not issue any cut commands and paper is fed through the presenter until a cut is issued
The minimum, maximum, and default values are shown in the Minimum, Maximum, and Default Settings table.

**Notes**

- If you are printing a multipage document with the setting **Cut at the document end** you get one long printout without a separation between each page.
- If you are printing a multipage document with the **Cut every page** setting each page is ejected with an ENQ (Clear Presenter) command after a cut if **Clear Presenter** is set to **Yes**.
- Use the **Cut at the document end** in connection with the **Partial Cut Width** setting to enable a document to be cut partially between pages and full at the end of a document.

**Partial cut width**

The **Partial cut width** setting affects the Printer Parameter 60 each time a print is issued. The possible values are between 1 and 40. The default value is 0. You need to set this value according to the print width of your printer (see the Technical Manual for more information on Parameter 60).

**Notes**

- The **Partial cut width** setting is only available for KR203, TTP 2000 and TTP 2100 series printers.
- Use the **Cut at the document end** in connection with the **Partial Cut Width** setting to enable a document to be partially cut between pages and fully cut at the end of a document.
- The **Partial cut width** setting cannot be used when the **Clear presenter** option is set to **Yes**.

**Example** • Partial cut width

You have a two page receipt that should be cut partially between the first and the second page. You are using an 80 mm paper and decide to cut 10 mm each side into the paper. You need to set the **Cutter mode** to **Cut at the document end** to indicate to the driver that you want only one full cut at the end of the document. Then select a **Partial cut width** of 10 allowing the printer to cut 10 mm into each side of the paper, and set the **Clear Presenter** to No. When you print your document the printer will print the first page, do a partial cut, print the second page, and do another partial cut, followed by a full cut. This is an expected behavior since neither the driver nor the printer knows the end of the document.

**Presenter loop length**

The **Presenter loop length** setting affects Printer Parameter 9 each time a print is issued. The minimum, maximum, and default values are shown in the Minimum, Maximum, and Default Settings table.
- **Eject length**
  The **Eject length** setting is affecting the physical length a ticket or receipt is ejected out of the presenter after a cut. The possible values are between 1 and 600 and represent the amount of media ejected in mm. (See the *Technical Manual* for more information on the ESC FF command.)

- **Present Length Addition**
  The **Present Length Addition** setting adds an additional amount to how far the paper is ejected during a present cycle. The possible values are between 1 and 255 and represent the amount of media ejected in mm. The default value is 0. (See the *Technical Manual* for more information on Parameter 47, Wall compensation.)

- **Presenter mode**
  The **Presenter mode** setting along with the **Presenter timeout** setting controls Printer Parameter 45. The two possible values are **Eject** (the default) and **Retract**. This takes effect when a new page is printed. (See the *Technical Manual* for further information on Parameter 45.)

  **Note** • This setting is not available for the KR203 or the TTP 2100 printer family.

- **Presenter timeout**
  The **Presenter timeout** setting along with the **Presenter mode** setting controls Printer Parameter 45. The possible values range from 0 to 300 and represent timeout delays in 10 second steps (e.g., a value of 3 is a 30 second timeout before the page is retracted into the waste bin).

  **Note** • Setting this value to 0 keeps the receipt in the presenter until the Kiosk user takes the receipt.

- **Clear presenter**
  The **Clear presenter** setting has two possible values and issues an ENQ (Clear Presenter) command if it is set to **Yes** or does nothing if it is set to **No** after the printer has cut and ejected a page. You can use this feature to fully eject a page from the presenter after it is printed.
• **Page hold**

The **Page hold** setting has two possible values. The driver holds a page in the presenter when printing a multipage document if the setting is set to **Yes**. Pages will not be held if the setting is set to **No**.

**Notes •**

• This feature only works if the **Enable bidirectional support** check box is checked in the **Ports** tab and the Language Monitor is running.

• This feature works together with the **Presenter mode** and **Presenter timeout** setting. If you do not allow retraction by setting the **Presenter timeout** value to **0** the print process hangs until the current page is taken out of the presenter because the driver does not send any new pages until the presenter has been cleared. If you allow retraction and the current page retracts due to the timeout period expiring while in hold mode the driver terminates the current print and no further pages print.

• Print jobs are held when the **Delete Print Job on Error** check box is cleared on the **Tools** tab. When printing one document multiple times it will still be looked at as one document in the print queue and deleted on error.

**Example • Page hold**

For this example, you want to print a multipage document and have **Presenter mode** set to **Retract**, **Presenter timeout** to 30 seconds (3), and the **Page hold** option to **Yes**.

When printing your document and taking every page out of the presenter before the timeout period expires, the driver sends each following page until the document is fully printed.

If printing your document and not taking a page and the timeout period expires the printer retracts this page and clears the presenter and also sends an error code to the driver indicating that a “retract” has occurred. The driver then stops printing and deletes the current print job.

The minimum, maximum, and default values are shown in the Minimum, Maximum, and Default Settings table.
**Driver Settings**

The **Driver Settings** enable you to adjust the contrast and brightness of the image. This feature is only available for the TTP 8000 series printers.

- **Image Adjustment**
  The **Image Adjustment** setting has two values: **Yes**, and **No**. If the value is set to Yes, the Contrast and Brightness settings are available. The default value is **No**.

- **Contrast**
  The **Contrast** settings minimum value is **-100**, the maximum value is **100**, and the default is **0**.

- **Brightness**
  The **Brightness** settings minimum value is **-100**, the maximum value is **100**, and the default is **0**.

The minimum, maximum, and default values are shown in the Minimum, Maximum, and Default Settings table.
Tools

The **Tools** tab allows you to clear the Kiosk Presenter, send a PRN file to the printer, feed a blank receipt, and print a configuration label that shows you the printer settings. You can also apply changes to various printer controls as described below.

- **Delete Print Job on Error**
  Print jobs are held when the **Delete Print Job on Error** check box is cleared on the **Tools** tab. When printing one document multiple times it will still be looked at as one document in the print queue and deleted on error. If you change this setting, you must click **Apply** and restart the spooler.

- **Send Printer Parameter**
  Select the **Send Printer Parameter** check box to send the driver settings from the **Device Settings** dialog to the printer. If you want to set your printer parameters manually without the driver overwriting them, clear the **Send Printer Parameter** check box.
• **Set vertical home position to zero**
  Select the **Set vertical home position to zero** to move the top margin of the printed image to the current print line location so that the printer is able to print the full image without cutting off the top portion of the image.

**Notes •**
- This does not affect the mechanical top margin of the printer, only the **Top margin** setting in the **Device Settings** dialog physically reverses the media.
- Click **Apply** for the changes to take effect.

**Printer Information**

The **Printer Information** tab shows the printer status for the selected printer.

**Note •** When you click **Refresh**, the Windows and Printer Error values are updated with the current status values. See **Windows Statuses** for a reference of the values.
**Import/Export settings**

The **Import/Export settings** tab enables you to export the driver settings to a file, import previously saved driver settings, and restore the printer to the factory default settings.

- **Import**
  Enables you to load a previously saved XML driver settings file. You should save the imported file in the C:\Zebra folder.

- **Export**
  Enables you to Export driver settings in a XML file. You can select which folder you want to use to save the file.

- **Default**
  Click **Default** to return the driver settings to the factory default.
Note • In Windows 7 Professional/Ultimate you need to start **Print Management** as an Administrator and select the printer that you want to Export, Import, or Default the Device Properties from.

If you default to the factory device settings you need to click **Cancel** to exit the **Printer Properties** dialog and reopen it to see the changes in the **Device Settings** tab. The values do not automatically refresh when you switch to the Device Setting tab.

![Print Management Screenshot](image)

**Note** • Click **Apply** for the changes to take effect.
About

The **About** tab shows the printer model and the driver version.
Printer Status Retrieval

Contents
The Language Monitor ................................................... 5 1
Windows APIs for Communication with the Printer .................... 5 1
Status Update in Windows “Printers and Faxes” or “Devices and Printer” .. 5 2

The Language Monitor

The Language Monitor is part of the Windows driver and is located between the Driver UI and the Port Monitor. The Language Monitor (LM) takes care of the direct communication with the selected port.

The Language Monitor has a Windows API interface through the GetPrinterData and GetPrinter functions.

All of the default Windows status responses can also be scripted with WMI scripts. See a description and a programming example in Status Monitoring & Programming Examples.

Windows APIs for Communication with the Printer

In order to make bi-directional communication easier and also compatible to more than one printer of the same kind on a specific PC, we implemented the LM GetPrinterData function. This is a Windows API described in Windows documentation. To retrieve immediate printer status from the Spooler you can also use the GetPrinter function. The GetPrinterData function is preferred over GetPrinter due to the fact that with GetPrinterData, all statuses and errors display, while with GetPrinter, only printer errors display.
• **GetPrinterData**
  

  **Note**
  
  • See *GetPrinterData Key Values* for available keywords.

• **GetPrinter**
  

  **Notes**
  
  • Zebra Printer status: It is recommended to use the PRINTER_INFO_3 structure to inquire for the printer status presented by the LM.
  
  • The spooler status is changed by `SetPort`. When using `SetPort` with custom messages, you cannot set these to be displayed or used by the spooler. This is a known bug: “SetPort doesn't work with custom status messages.” (*Microsoft*) Therefore, all custom messages will be declared as PRINTER_STATUS_NOT_AVAILABLE and a KPL value is placed in the `ExternalError` key. The custom messages are only accessible through the `GetPrinterData` function.

### Status Update in Windows “Printers and Faxes” or “Devices and Printer”

In the case that the printer is not printing the status will be checked every 1.5 seconds (depending on the setting of the `READ_THREAD_IDLE_SLEEP` key in the LM registry setting). During printing and on error the status will be checked more frequently.

The same status that can be gathered with the `GetPrinterData` or `GetPrinter` API will be displayed in the Printer folder.

  **Note**
  
  In some cases it may be possible that the PnP ping is not properly executed on the system and therefore the idle thread of the LM is not activated after a power off situation of the printer. In this case the LM is reactivated the next time a print job is executed.
Windows Statuses

Contents
Windows Compatible Status .............................................. 53
Windows Incompatible Status ............................................. 55

Windows Compatible Status
These statuses appear in the Printers and Faxes dialog and are stored in the printer ERROR key in the Registry. They can be extracted with GetPrinterData.

Statuses Defined in winspool.h

Table 1 • Windows Status Compared to Zebra Status

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRINTER_STATUS_PAPER_JAM</td>
<td>Paper jam (ESC ENQ 1 = NAK 1)</td>
</tr>
<tr>
<td>PRINTER_STATUS_USER_INTERVENTION</td>
<td>Cutter not home (ESC ENQ 1 = NAK 2)</td>
</tr>
<tr>
<td>PRINTER_STATUS_PAPER_OUT</td>
<td>Out of paper (ESC ENQ 1 = NAK 3)</td>
</tr>
<tr>
<td>PRINTER_STATUS_DOOR_OPEN</td>
<td>Print head lifted (ESC ENQ 1 = NAK 4)</td>
</tr>
<tr>
<td>PRINTER_STATUS_PAPER_PROBLEM</td>
<td>Paper feed problem (ESC ENQ 1 = NAK 5)</td>
</tr>
<tr>
<td>PRINTER_STATUS_NOT_AVAILABLE</td>
<td>Temperature error (ESC ENQ 1 = NAK 6)</td>
</tr>
<tr>
<td>PRINTER_STATUS_ERROR</td>
<td>Presenter jam (ESC ENQ 1 = NAK 7), check ExternalError</td>
</tr>
<tr>
<td>PRINTER_STATUS_PAPER_JAM</td>
<td>Retract jam (ESC ENQ 1 = NAK 8), check ExternalError</td>
</tr>
<tr>
<td>PRINTER_STATUS_NOT_AVAILABLE</td>
<td>Black mark not found (ESC ENQ 1 = NAK 10), check ExternalError</td>
</tr>
<tr>
<td>PRINTER_STATUS_NOT_AVAILABLE</td>
<td>Black mark calibration error (ESC ENQ 1 = NAK 11), check ExternalError</td>
</tr>
<tr>
<td>PRINTER_STATUS_NOT_AVAILABLE</td>
<td>Index error (ESC ENQ 1 = NAK 12), check ExternalError</td>
</tr>
</tbody>
</table>
## Table 1 • Windows Status Compared to Zebra Status

<table>
<thead>
<tr>
<th>Windows Status</th>
<th>Zebra Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRINTER_STATUS_NOT_AVAILABLE</td>
<td>Checksum error (ESC ENQ 1 = NAK 13), check ExternalError</td>
</tr>
<tr>
<td>PRINTER_STATUS_NOT_AVAILABLE</td>
<td>Wrong firmware (ESC ENQ 1 = NAK 14), check ExternalError</td>
</tr>
<tr>
<td>PRINTER_STATUS_NOT_AVAILABLE</td>
<td>Retract occurred (ESC ENQ 1 = NAK 16), check ExternalError</td>
</tr>
<tr>
<td>PRINTER_STATUS_NOT_AVAILABLE</td>
<td>Paused (ESC ENQ 1 = NAK 17), check ExternalError</td>
</tr>
<tr>
<td>PRINTER_STATUS_Toner_Low</td>
<td>Paper near end (ESC ENQ 6)</td>
</tr>
<tr>
<td>PRINTER_STATUS_No_Toner</td>
<td>Weekend paper status (ESC ENQ 6) (only for TTP 7030 and TTP 8000 with special hardware)</td>
</tr>
<tr>
<td>PRINTER_STATUS_Output_Bin_Full</td>
<td>Paper in presenter (ESC ENQ 6)</td>
</tr>
</tbody>
</table>

## Table 2 • Status definition in Winspool.h

```c
#define PRINTER\_STATUS\_ERROR 0x00000002
#define PRINTER\_STATUS\_PAPER\_JAM 0x00000008
#define PRINTER\_STATUS\_PAPER\_OUT 0x00000010
#define PRINTER\_STATUS\_PAPER\_PROBLEM 0x00000040
#define PRINTER\_STATUS\_OFFLINE 0x00000080
#define PRINTER\_STATUS\_OUTPUT\_BIN\_FULL 0x00000800
#define PRINTER\_STATUS\_NOT\_AVAILABLE 0x00001000
#define PRINTER\_STATUS\_TONER\_LOW 0x00020000
#define PRINTER\_STATUS\_NO\_Toner 0x00040000
#define PRINTER\_STATUS\_USER\_INTERVENTION 0x00100000
#define PRINTER\_STATUS\_DOOR\_OPEN 0x00400000
```

**Note** • In order to indicate the Kiosk printer status of Paper-near-end or Weekend-paper-status Zebra is using two Microsoft status codes that are not used by thermal printers, as they do not need any toner. The codes used are PRINTER\_STATUS\_TONER\_LOW for Paper-near-end and PRINTER\_STATUS\_NO\_Toner for Weekend-paper-status. These statuses are only informative and do not block printing. The Weekend-paper-status is only present with printers that have the option of two sensors on their roll holder. (See the *Technical Manual* for your printer for more information on the available options.)
Windows Incompatible Status

If a printer status doesn’t have a corresponding Windows status the Error key will have PRINTER_STATUS_NOT_AVAILABLE set and you need to evaluate the ExternalError key.

Statuses that have a representation within the Windows status may also have an ESC ENQ 1 NAK value (see Table 4) and will be stored in the printer ExternalError key in the registry and can be extracted with GetPrinterData using the ExternalError key.

For the meanings of these NAK responses see the appropriate Technical Manual for your printer, under the ESC ENQ 1 section.

Note • Any other Windows status may be used in the future, so mask away undefined bits in your application!
### GetPrinterData Key Values

#### Contents

GetPrinterData Key Values. ................................................ 57

---

#### GetPrinterData Key Values

<table>
<thead>
<tr>
<th>Printer</th>
<th>DsMonitor Key Explanation</th>
<th>Type</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>Printer Error or Status in Windows 16-bit format</td>
<td>REG_DWORD</td>
<td></td>
</tr>
<tr>
<td>ErrorEvent</td>
<td>Error event name for error event trigger</td>
<td>REG_SZ</td>
<td>only in Windows XP</td>
</tr>
<tr>
<td>ExternalError</td>
<td>Extended status according to Appendix B</td>
<td>REG_DWORD</td>
<td></td>
</tr>
<tr>
<td>Firmware</td>
<td>Firmware version</td>
<td>REG_BINARY</td>
<td>only for USB connection</td>
</tr>
<tr>
<td>PageCount</td>
<td>Page counter for cut pages</td>
<td>REG_DWORD</td>
<td></td>
</tr>
<tr>
<td>PCB_REV</td>
<td>Printers PCB revision number</td>
<td>REG_BINARY</td>
<td></td>
</tr>
<tr>
<td>PCB_SN</td>
<td>Printers PCB serial number</td>
<td>REG_BINARY</td>
<td></td>
</tr>
<tr>
<td>StatusEvent</td>
<td>Status event name for status event trigger</td>
<td>REG_SZ</td>
<td>only for Windows XP</td>
</tr>
<tr>
<td>RetractCount</td>
<td>Retract counter for retracted pages</td>
<td>REG_DWORD</td>
<td></td>
</tr>
<tr>
<td>DeleteJob</td>
<td>Flag to delete print jobs on error</td>
<td>REG_DWORD</td>
<td></td>
</tr>
<tr>
<td>Head_Temp</td>
<td>Head temperature (ESC ENQ B)</td>
<td>REG_DWORD</td>
<td></td>
</tr>
</tbody>
</table>
Status Monitoring

In order to incorporate the new way of status monitoring you need some background on what happens in a Kiosk when you print and when you should monitor your status.

Status monitoring can be handled in two different ways.

- Monitor in your printing application
- Monitor in a separate application

When you monitor in your printing application you would commonly look at the printer before sending a print job to see if the printer is OK and then send your print job. After the print job is signaled as being printed you would check status again to see if the printer has any errors or if the paper has been taken, etc.

Monitoring in a separate application usually doesn’t allow direct interaction with the printed job so you are trying to poll the printer as often as you can to get the most accurate information on what the printer is doing. This is usually a very time consuming task and you have to care for synchronizing with a current print job.

Since monitoring in a separate application is most commonly used for status monitoring, we have incorporated an event notification into the Language Monitor (LM) to allow a monitoring application to do other tasks and have a separate thread listening for the printer status or error event change. When this occurs the thread is simply getting the status and reporting this back to the main program or doing any other kind of reporting.
To accommodate this notification for all error and status changes we incorporated two mechanisms in the LM.

**Monitoring while printing**

We implemented status monitoring in the internal printing structure of the LM. When you open a Document, print it and close the Document again the LM will check the printer status before and after printing and will also react to write errors if such occur. Then it will set the printer status and raise the error event.

**Monitoring while idle**

We implemented an internal status thread which polls the printer when it is idle in a predefined cycle and provides changed status information in the same manner. It will set the status and raise an error or status event. Therefore, it is not necessary to implement your own monitoring loop. You can simply wait for an event in your application’s idle loop.

**Implementation in Calling Application**

**Note** • The following example is not applicable for Windows 7 and above.

1. Open the Printer.

   The first step of your implementation is to open the printer you want to monitor and get the Error event and Status event name.

   ```
   bRet = OpenPrinter(m_csPrinter.GetBuffer(1), &hPrinter, &pd);
   ...
   if ((dRet = GetPrinterData(hPrinter, "ErrorEventName", &dType, (LPBYTE)cTmp, 100, &dNeeded))!=ERROR_SUCCESS)
   ...
   if ((dRet = GetPrinterData(hPrinter, "StatusEventName", &dType, (LPBYTE)cTmp, 100, &dNeeded))!=ERROR_SUCCESS)
   ...
   ```

2. Open the Event Handles.

   Open the two event handles and fill these handles into a structure you will pass on to the new thread.

   ```
   typedef struct _CStatusThreadInfo
   {
     HWNDmyHwnd;
     DWORDdSleepTime;
     HANDLEhPrinter;
     HANDLEhError;
     HANDLEhStatus;
     BOOLm_hStatusEventKillThread;
   } CStatusThreadInfo;
   ```

   ```
   ...
Implementation in Monitor Thread for OS Prior to Windows 7

1. Fill Event Arrays
   In the monitoring thread you create and fill an array of handles with the error and status event handle.
   
   ```
   myHandle[0] = pInfo->hError;
   myHandle[1] = pInfo->hStatus;
   ```

2. Start the Waiting Loop
   Then you are ready to start the waiting loop.
   
   ```
   for ( ; ; )
   {
   if (pInfo->m_hStatusEventKillThread)
   {
   OutputDebugStringA("### [Thread msg.] Kill thread...
   pInfo->m_hStatusEventKillThread = FALSE;
   AfxEndThread( 1 );
   return 1;
   }
   if ((dwRet = WaitForMultipleObjects(2, myHandle, FALSE, pInfo->dSleepTime))!=WAIT_FAILED)
   {
   if (dwRet==WAIT_OBJECT_0 || dwRet==WAIT_OBJECT_0+1)
   {
   if ((dwRet = GetPrinterData(hPrinter, "Error", &dType, (LPBYTE)&dwResult, sizeof(dwResult), &dNeeded))!=ERROR_SUCCESS)
   {
   sprintf( str, "### [Status Thread error %d] read [%08X]\n", dwRet, dwResult);
   OutputDebugStringA(str);
   
   // handle other status errors
   
   // handle error in waiting loop
   
   // handle error in thread function
   ```
SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_ERROR"));
    if (dwResult & PRINTER_STATUS_PENDING_DELETION)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PENDING_DELETION"));
    if (dwResult & PRINTER_STATUS_PAPER_JAM)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_JAM"));
    if (dwResult & PRINTER_STATUS_PAPER_OUT)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_OUT"));
    if (dwResult & PRINTER_STATUS_PAPER_PROBLEM)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_PROBLEM"));
    if (dwResult & PRINTER_STATUS_OFFLINE)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_OFFLINE"));
    if (dwResult & PRINTER_STATUS_IO_ACTIVE)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_IO_ACTIVE"));
    if (dwResult & PRINTER_STATUS_BUSY)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_BUSY"));
    if (dwResult & PRINTER_STATUS_PRINTING)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PRINTING"));
    if (dwResult & PRINTER_STATUS_OUTPUT_BIN_FULL)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_OUTPUT_BIN_FULL"));
    if (dwResult & PRINTER_STATUS_PROCESSING)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PROCESSING"));
    if (dwResult & PRINTER_STATUS_USER_INTERVENTION)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_USER_INTERVENTION"));
    if (dwResult & PRINTER_STATUS_DOOR_OPEN)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_DOOR_OPEN"));
    if (dwResult & PRINTER_STATUS_TONER_LOW)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_TONER_LOW"));
    if (dwResult & PRINTER_STATUS_NO_TONER)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_NO_TONER"));
    if (dwResult & PRINTER_STATUS_PAPER_NEAR_END)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_NEAR_END"));
    if (dwResult & PRINTER_STATUS_EXTERNAL_ERROR)
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_EXTERNAL_ERROR"));
    if (((dwRet = GetPrinterData(hPrinter, "ExternalError", &dType, (LPBYTE)dwResult, sizeof(dwResult), &dNeeded))!=ERROR_SUCCESS)
        sprintf( str, "### [Status Thread error %d] read [%08X]n", dwRet, dwResult);
        OutputDebugStringA(str);
When an event occurs you need to get the status with GetPrinterData using the “Error” key and decode the result according to the sample or any way you feel necessary. In any case you can send a message or do any form of status reporting you want to do.

Implementation in Monitor Thread for OS Windows 7 and Above

Status function called from within the monitoring thread:

```c
void getStatus(LPVOID pParam) {
    CStatusThreadInfo* pInfo = (CStatusThreadInfo*)pParam;
    char str[150];
    long dwResult=0, dwRet=0;
    DWORD dType, dNeeded;
    DWORD dwPgCnt=0;
    DWORD dwCutCnt=0;

    // Get status from the Language Monitor
    if ((dwRet = GetPrinterData(hPrinter, "Error", &dType, (LPBYTE)&dwResult, sizeof(dwResult), &dNeeded))!=ERROR_SUCCESS) {
        sprintf_s(str, 150, "### Wait function failed! [%d]\n", dwRet);
        OutputDebugStringA(str);
    }
    else {
        dwRet = GetLastError();
        sprintf_s(str, 150, "### [Status Thread External Error] read [%08X]\n", dwResult);
        OutputDebugStringA(str);
        SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)(str));
    }

    // When the status is ready, read it
    if (dwRet != ERROR_SUCCESS) {
        if (dType==ERROR) {
            if (dNeeded) {
                dType = (DWORD)dwNeeded;
                SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)(str));
            }
            else {
                SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("Timeout"));
            }
        }
    }
}
```

When an event occurs you need to get the status with GetPrinterData using the “Error” key and decode the result according to the sample or any way you feel necessary. In any case you can send a message or do any form of status reporting you want to do.
SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)(str));

// parse the status result and output the result
if (dwResult & 0x00000000)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_OK");
if (dwResult & PRINTER_STATUS_ERROR)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_ERROR");
if (dwResult & PRINTER_STATUS_PENDING_DELETION)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PENDING_DELETION");
if (dwResult & PRINTER_STATUS_PAPER_JAM)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_JAM");
if (dwResult & PRINTER_STATUS_PAPER_OUT)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_OUT");
if (dwResult & PRINTER_STATUS_PAPER_PROBLEM)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_PROBLEM");
if (dwResult & PRINTER_STATUS_OFFLINE)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_OFFLINE");
if (dwResult & PRINTER_STATUS_IO_ACTIVE)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_IO_ACTIVE");
if (dwResult & PRINTER_STATUS_BUSY)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_BUSY");
if (dwResult & PRINTER_STATUS_PROCESSING)
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PROCESSING");
if (dwResult & PRINTER_STATUS_USER_INTERVENTION) {
    dwCutCnt++;
    sprintf_s( str, 150, "[%d]", dwCutCnt);
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_CUT_ERR), WM_SETTEXT, 0, (LPARAM)str);
}
if (dwResult & PRINTER_STATUS_DOOR_OPEN)
SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_DOOR_OPEN"));

if ((dwResult & PRINTER_STATUS_PAPER_NEAR_END)||(dwResult & PRINTER_STATUS_TONER_LOW))
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_NEAR_END"));
if ((dwResult & PRINTER_STATUS_PAPER_WEEKEND)||(dwResult & PRINTER_STATUS_NO_TONER))
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_WEEKEND"));
if ((dwResult & PRINTER_STATUS_PAPER_PRESENTER) || (dwResult & PRINTER_STATUS_OUTPUT_BIN_FULL)) {
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_PAPER_PRESENTER"));
    dwPgCnt++;
    sprintf_s( str, 150, "]%d]\n", dwPgCnt);
    SendMessage(GetDlgItem(HWND)pInfo->myHwnd, IDC_PG_CNT), WM_SETTEXT, 0, (LPARAM)str);
}
// if we have more information available about the specific error get it now
if ((dwResult & PRINTER_STATUS_EXTERNAL_ERROR) || (dwResult & PRINTER_STATUS_USER_INTERVENTION) || (dwResult & PRINTER_STATUS_NOT_AVAILABLE))
{
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)("PRINTER_STATUS_EXTERNAL_ERROR"));
    if ((dwRet = GetPrinterData(hPrinter, "EXTERNALERROR", &dType, (LPBYTE)&dwResult, sizeof(dwResult), &dNeeded))!=ERROR_SUCCESS)
    {
        sprintf_s( str, 150, "]Status Thread error %d] read [\%08X]\n", dwRet, dwResult);
        OutputDebugStringA(str);
    }
    sprintf_s( str, 150, "]Status Thread External Error] read [\%08X]\n", dwResult);
    OutputDebugStringA(str);
    SendMessage(GetDlgItem((HWND)pInfo->myHwnd, IDC_Status), WM_SETTEXT, 0, (LPARAM)str);
}
}
Status Monitoring Thread

Note • See Implementation in Monitor Thread for OS Prior to Windows 7 for implementation with event handling.

UINT StatusThreadProc( LVOID pParam)
{
    CStatusThreadInfo* pInfo = (CStatusThreadInfo*)pParam;
    long dwResult=0, dwRet=0;
    char str[150];
    HANDLE myHandle[2];
    DWORD dwPgCnt=0;
    DWORD dwCutCnt=0;

    if (pInfo == NULL)
    {
        OutputDebugStringA("### entering Status Poll thread Failed!\n");
        return 1;   // if pObject is not valid
    }

    OutputDebugStringA("### entering Status Poll thread...\n");
    BOOL isWin7 = IsWin7();

    for ( ; ; )
    {
        if (pInfo->m_hStatusEventKillThread)
        {
            OutputDebugStringA("### [Thread msg.] Kill thread...\n");
            pInfo->m_hStatusEventKillThread = FALSE;
            AfxEndThread( 1 );
            return 1;
        }
        // in Windows 7 and above the event system used in XP is broken and the status has to
        // be acquired through polling
        if (isWin7)
        {
            getStatus(pParam);
            Sleep(pInfo->dSleepTime);
        }
    }
    return 1;
}
Status Implementation with C#

Prerequisite is to declare all the Win API stuff for C#:

```csharp
class PrintSpoolerApi
{
    [DllImport("winspool.drv", SetLastError = true, CharSet = CharSet.Auto)]
    public static extern bool OpenPrinter(
        [MarshalAs(UnmanagedType.LPTStr)]
        string printerName,
        out IntPtr printerHandle,
        PrinterDefaults printerDefaults);

    [DllImport("winspool.drv", SetLastError = true, CharSet = CharSet.Ansi,
        CallingConvention = CallingConvention.StdCall)]
    public static extern UInt32 GetPrinterData(
        IntPtr hPrinter,
        [MarshalAs(UnmanagedType.LPStr)]
        string pValueName,
        ref uint pType,
        ref UInt32 pData,
        uint nSize,
        ref uint pcbNeeded );

    [DllImport("winspool.drv", SetLastError = true, CharSet = CharSet.Auto)]
    public static extern bool GetPrinter(
        IntPtr printerHandle,
        int level,
        IntPtr printerData,
        int bufferSize,
        ref int printerDataSize);

    [DllImport("winspool.drv", SetLastError = true, CharSet = CharSet.Auto)]
    public static extern bool ClosePrinter(
        IntPtr printerHandle);

    [StructLayout(LayoutKind.Sequential)]
    public struct PrinterDefaults
    {
        public IntPtr pDatatype;
        public IntPtr pDevMode;
        public int DesiredAccess;
    }
}
```
public enum PrinterProperty
{
    ServerName,
    PrinterName,
    ShareName,
    PortName,
    DriverName,
    Comment,
    Location,
    PrintProcessor,
    Datatype,
    Parameters,
    Attributes,
    Priority,
    DefaultPriority,
    StartTime,
    UntilTime,
    Status,
    Jobs,
    AveragePpm
};

public struct PrinterInfo2
{
    [MarshalAs(UnmanagedType.LPTStr)]
    public string ServerName;
    [MarshalAs(UnmanagedType.LPTStr)]
    public string PrinterName;
    [MarshalAs(UnmanagedType.LPTStr)]
    public string ShareName;
    [MarshalAs(UnmanagedType.LPTStr)]
    public string PortName;
    [MarshalAs(UnmanagedType.LPTStr)]
    public string DriverName;
    [MarshalAs(UnmanagedType.LPTStr)]
    public string Comment;
    [MarshalAs(UnmanagedType.LPTStr)]
    public string Location;
    public IntPtr DevMode;
    [MarshalAs(UnmanagedType.LPTStr)]
    public string SepFile;
    [MarshalAs(UnmanagedType.LPTStr)]
    public string PrintProcessor;
}
Getting status with GetPrinterData using C#

public static unsafe UInt32 GetPrinterStatus(string printerUncName)
{
    var pHandle = new IntPtr();
    var defaults = new PrinterDefaults();
    string StatusKey="Error";
    uint pType=0;
    byte[] pData=new byte[255];
    uint pcbNeeded=0;

    try
    {
        //Open a handle to the printer
        bool ok = OpenPrinter(printerUncName, out pHandle, defaults);

        if (!ok)
        {
            //OpenPrinter failed, get the last known error and thrown it
            throw new Win32Exception(Marshal.GetLastWin32Error());
        }

        //Here we determine the size of the data we to be returned
        //Passing in 0 for the size will force the function to return the size of the data requested
        //byte* pbData = &pData[0];
        UInt32 pbData = 0;
        UInt32 nVal = 0;

        nVal = GetPrinterData(pHandle, StatusKey, ref pType, ref pbData, (uint)4, ref pcbNeeded);
    }

    //Close the handle
    CloseHandle(pHandle);
}
int err = Marshal.GetLastWin32Error();

if (err == 122)
{
    if (nVal > 0)
    {
        //Allocate memory to the size of the data requested
        pData = new byte[pcbNeeded];
        //Retrieve the actual information this time
        nVal = GetPrinterData(pHandle, StatusKey, ref pType, ref pbData,
        (uint)pData.Length, ref pcbNeeded);
        err = Marshal.GetLastWin32Error();
        return pbData;
    }
    return pbData;
}
finally
{
    //Always close the handle to the printer
    ClosePrinter(pHandle);
}

Using GetPrinter
public static PrinterInfo2 GetPrinterProperty(string printerUncName)
{
    var printerInfo2 = new PrinterInfo2();

    var pHandle = new IntPtr();
    var defaults = new PrinterDefaults();
    try
    {
        //Open a handle to the printer
        bool ok = OpenPrinter(printerUncName, out pHandle, defaults);

        if (!ok)
        {
            //OpenPrinter failed, get the last known error and throw it
            throw new Win32Exception(Marshal.GetLastWin32Error());
        }

        //Here we determine the size of the data we to be returned
/Passing in 0 for the size will force the function to return the size of the data requested
int actualDataSize = 0;
GetPrinter(pHandle, 2, IntPtr.Zero, 0, ref actualDataSize);

int err = Marshal.GetLastWin32Error();

if (err == 122)
{
    if (actualDataSize > 0)
    {
        //Allocate memory to the size of the data requested
        IntPtr printerData = Marshal.AllocHGlobal(actualDataSize);
        //Retrieve the actual information this time
        GetPrinter(pHandle, 2, printerData, actualDataSize, ref actualDataSize);

        //Marshal to our structure
        printerInfo2 = (PrinterInfo2)Marshal.PtrToStructure(printerData, typeof(PrinterInfo2));
        //We've made the conversion, now free up that memory
        Marshal.FreeHGlobal(printerData);
    }
}
else
{
    throw new Win32Exception(err);
}

return printerInfo2;
}
finally
{
    //Always close the handle to the printer
    ClosePrinter(pHandle);
}
WMI Script to get Basic Status

' VBScript source code
ttpname=""
strComputer = "."
Set objWMIService = GetObject("winmgmts:" _
   & "{impersonationLevel=impersonate}!\" & strComputer & "\root\cimv2")
Set wbemObjectSet = objWMIService.ExecQuery("SELECT * FROM Win32_Printer")
For Each wbemObject In wbemObjectSet
    if wbemObject.Default = TRUE then
        ttpname = wbemObject.Caption
        Wscript.Echo "Printer " & ttpname
        Select Case wbemObject.PrinterStatus
            Case 1
                strPrinterStatus = "Other"
                strExtendedPrinterStatus = wbemObject.ExtendedPrinterStatus
            Case 2
                strPrinterStatus = "Unknown"
            Case 3
                strPrinterStatus = "Idle"
            Case 4
                strPrinterStatus = "Printing"
            Case 5
                strPrinterStatus = "Warmup"
            Case 6
                strPrinterStatus = "Stopped printing"
            Case 7
                strPrinterStatus = "Offline"
        End Select
        Wscript.Echo "Printer Status: " & strPrinterStatus
        Select Case wbemObject.DetectedErrorState
            Case 0
                Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Unknown"
            Case 1
                Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Other"
            Case 2
                Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " No Error"
            Case 3
                Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Low Paper"
            Case 4
    End If
End For
case 5
    Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Low Toner"
case 6
    Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " No Toner"
case 7
    Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Door Open"
case 8
    Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Jammed"
case 9
    Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Offline"
case 10
    Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Service Requested"
case 11
    Wscript.Echo "DetectedErrorState: " & wbemObject.DetectedErrorState & " Output Bin Full"
End Select
Select Case wbemObject.ExtendedDetectedErrorState
    Case 0
        Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Unknown"
case 1
        Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Other"
case 2
        Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " No Error"
case 3
        Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Low Paper"
case 4
        Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " No Paper"
case 5
        Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Low Toner"
case 6
        Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " No Toner"
case 7
        Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Door Open"
case 8
Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Jammed"
case 9
  Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Service Requested"
case 10
  Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Output Bin Full"
case 11
  Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Paper Problem"
case 12
  Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Cannot Print Page"
case 13
  Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " User Interantion Required"
case 14
  Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Out of Memory"
case 15
  Wscript.Echo "ExtendedDetectedErrorState: " & wbemObject.ExtendedDetectedErrorState & " Server Unknown"
End Select
end if
Next
Wscript.Echo "Printer " & ttpname
Setup Print Forms in Windows XP and Vista

Windows XP and Vista allows you to control global settings for print servers by using the Print Server Properties dialog. You can access this dialog by doing the following:

1. Double-click on the printer’s icon in the Control Panel or select Settings in the Start menu and then choose the Printers option.

2. In the Printers window, select Server Properties from the File menu.
3. Use the Forms tab of the Print Server Properties dialog to view printer forms.

**Viewing and Creating Print Forms**

Forms are used by the print server to define the standard sizes for paper, envelopes, and transparencies. To view the current settings for a printer form, follow these steps:

1. Open the Print Server Properties dialog and then click on the Forms tab as shown above.

2. Use the Forms On list box to select the form you want to view.

   The form settings are shown in the Measurements area. You can't change or delete the default system forms.

**To create a new form, follow these steps:**

- **Note** • You must give the form you create a new name to ensure that the original form remains usable.

1. Access the Forms tab of the Print Server Properties dialog.

2. Use the Forms On list box to select the existing form on which you want to base the new form.

3. Select the Create A New Form check box.

4. Enter a new name for the Form in the Form Description For field.
5. Use the fields in the **Measurements** area to set the paper size and margins.

6. Click the **Save Form** button to save the form. Give the form a new name to ensure that the original form remains usable.

**Setup Print Forms in Windows 7**

You can use **Print Management** to manage print forms.

**Note** • You must be signed in as an Administrator to use Print Management.

1. To open Print Management, type `printmanagement.msc` in the search box, and then press **Enter**.

2. Open **Print Management**.
3. In the left pane, click **Print Servers**, click the applicable print server, right-click **Forms**, and then click **Manage Forms**.

In the **Print Server Properties** dialog, do the following steps.

4. To create a new form, select an existing form, select the **Create a new form** check box, change the printer measurement units, paper size, and printer area margins as needed, click **Save Form**, and then click **OK**.
Note • You must give the form you create a new name to ensure that the original form remains usable.

5. To delete a form, select the form, click **Delete**, and then click **OK**.

**Additional References**

- How to find PaperSize for custom print sizes under Windows NT and later versions by using Windows API functions
  
  [http://support.microsoft.com/kb/304639](http://support.microsoft.com/kb/304639)
  
  Article ID: 304639 - Last Review: February 2, 2005 - Revision: 4.4

- Manage Forms in Windows 7 and Server 2008 R2
  

- Configuring Print Server Properties in Windows XP and Vista
  
A
About tab, 50
APIs, 51

B
Bottom margin, 40, 41
Brightness, 45

C
Clear presenter, 43
continuous, 40
Contrast, 45
copy count, 26
Cut at the document end, 41
Cut every page, 41
Cutter mode, 41

D
Darkness, 39
Default, 48
Delete Print Job on Error, 46
Document Settings, 40
driver installation, 16
Driver Settings, 45

E
Eject length, 43
Export, 48

F
Form To Tray Assignment, 35

G
General tab, 24

I
Image Adjustment, 45
Import, 48
Import/Export settings tab, 48

L
Language Monitor, 51
Layout tab, 25
liability, 2

M
mark sensing, 40
Max print speed, 39
Media Tracking, 40

N
No Cut, 41

O
OEM, 22

P
Page hold, 44
paper size, 26
Paper/Quality tab, 25
Partial cut width, 42
Present Length Addition, 43
Presenter loop length, 42
Index

Presenter mode, 43
Presenter Settings, 41
Presenter timeout, 43
Print Management, 14
Printer Information tab, 47
Printer Settings, 39
Properties dialog, 23

S
Send Printer Parameter, 46
Set vertical home position to zero, 47

T
Tools tab, 46
Top margin, 40

U
uninstall, 9

V
variable length, 40

W
Windows 7, 14
Windows XP, 10