



# **PEERNET® Reports Sample J2EE Solution**

**Version 2.6**

## **User Guide**

**Revised: 01 May 2007**

<http://www.peernet.com/reports>

---

<b>Chapter 1: About this User Guide</b> .....	<b>3</b>
What's in this User Guide? .....	3
System requirements .....	3
Technical support .....	4
<b>Chapter 2: PEERNET Reports Sample J2EE Solution</b> .....	<b>5</b>
Creating PEERNET Reports projects .....	5
Deploying PEERNET Reports projects .....	5
<b>Chapter 3: Working with PEERNET Reports Sample J2EE Solution</b> .....	<b>7</b>
Using PEERNET Reports Server .....	7
Printing reports and labels from a web page .....	8
Embedding report or label images into a web page.....	8
Linking report or label PDF documents to a web page .....	8
Working with the PEERNET Reports Applet .....	8
PEERNET Reports Applet .....	11

---

## Chapter 1: About this User Guide

This User Guide is intended to guide you through the setup and use of PEERNET Reports Sample J2EE Solution.

### What's in this User Guide?

This User Guide is divided into the following chapters:

- About this User Guide
- PEERNET Reports Sample J2EE Solution
- Working with PEERNET Reports Sample J2EE Solution

For a complete listing of what's available in this User Guide, please refer to the Table of Contents. Note that you can download the latest edition of this User Guide from the PEERNET website.

### System requirements

The following are the minimum system requirements for this application:

256 MB RAM  
200 MB hard disk space  
Monitor with 800 x 600 resolution  
Mouse or other pointing device

#### Windows®

Intel® Pentium® II/233 MHz or higher  
Java® Virtual Machine 1.4 or higher (included with product)

#### Linux®

Intel Pentium II/233 MHz or higher  
Red Hat® Linux 6.2 or 7.2 with default GNOME or KDE desktop managers  
Java Virtual Machine 1.4 or higher (included with product)  
XVFB or X Windows

#### Solaris™

UltraSPARC® II or higher  
Solaris 7 or higher  
Java Virtual Machine 1.4 or higher (included with product)  
XVFB or X Windows

---

## Technical support

For information on obtaining technical assistance with this product, please visit the product-specific support area of our web site at <http://www.peernet.com/support>.

PEERNET offers unlimited email support for 60 days from date of purchase of this product. Send your support issues via email to [support@peernet.com](mailto:support@peernet.com). Please include in your message:

- the serial number of your product
- the type of operating system you are using
- the application you are using
- a detailed description of any error messages you have received
- a detailed description of the steps you took when the problem occurred
- your daytime telephone number, in case we need to call you

Our technical personnel are available to answer your questions from 08h30 to 16h30, Monday to Friday, Eastern Standard Time.

---

## Chapter 2: PEERNET Reports Sample J2EE Solution

PEERNET Reports is a revolutionary reporting and labeling product that enables you to create and print reports or labels (with extensive support for bar coding included) as part of your web-based applications. Sample uses for this product include invoices, manifests, catalogs, event tickets, shipping labels, and pre-printed stock labels; the possibilities are limitless.

There are two ways in which PEERNET Reports Sample J2EE Solution can be utilized.

First, you can deploy PEERNET Reports Server (a Web Application) onto your existing Web Application Server (WebLogic, JRun, Tomcat, Jakarta, etc.). In this case, the Web Application file **PEERNETReportsServer2.war**, located in the **redist** folder where you installed the product, should be deployed to your Web Application Server. Performing this task will make the servlet **PEERNETReportsServerRequestServlet**, as well as the web pages used for licensing and managing PEERNET Reports Server, available for use,

Second, you can embed the PEERNET Reports runtime engine into a new or existing system using the PEERNET Reports Object Model Java API (a callable Java class library). For complete information about using the Embedded Engine, refer to the **PEERNET Reports Embedded Engine User Guide**.

### Creating PEERNET Reports projects

PEERNET Reports Designer provides the visual tools to create and manage all aspects of your reporting or labeling projects. For complete information about using PEERNET Reports Designer, refer to the **PEERNET Reports Designer User Guide**.

### Deploying PEERNET Reports projects

The instructions below describe how to deploy a PEERNET Reports project to the PEERNET Reports Server. Note that, if you are using the Embedded Engine directly, your deployment method will be similar, but it will be based on how you have embedded the product into your application.

PEERNET Reports Server is a Web Application, and should be installed using the default name **PEERNETReportsServer2**. In the **webapps** folder you will see a **PEERNETReportsServer2** folder, and it will feature the following structure:

```
PEERNETReportsServer2
  \applet
  \images
  \META-INF
  \WEB-INF
    \classes
    \lib
    \projects
      \examples
        \labels
        \Northwind
    \fonts
```

In the base PEERNET Reports for J2EE installation an evaluation environment has been provided for PEERNET Reports Server using the Java Application Server **tomcat**. The **webapps** folder will be at **<installdir>\PEERNETReports2.0\jakarta-tomcat-3.3.1\webapps** on a Windows platform, or at **<installdir>/PEERNETReports2.0/jakarta-tomcat-3.3.1/webapps** on the other supported platforms.

---

There are two key folders in the **WEB-INF** folder: **projects** and **fonts**. To deploy a PEERNET Reports project, you must copy the project (PNJ) file to the **projects** folder. You might also need to copy any TrueType font files referenced by your reports or labels to the **fonts** folder. On a Windows platform, this is not a requirement if **PEERNETOS2Extension.dll** has been added to the PATH variable during the startup of the Java Application Server. Note that, in the base installation, we do not add the **PEERNETOS2Extension**; this is to demonstrate the requirement that fonts must be copied to the **fonts** folder.

From the PEERNET Reports Server Web Application home page (which is located at the address of **<http://<webserver>:<port>/PEERNETReports2/index.htm>**), you can retrieve a list of the deployed projects, and also retrieve a list of the fonts available for use by any deployed project. For the base installation, once you have started the J2EE Server, you can access the home page at **<http://localhost:8080/PEERNETReports2/index.htm>** or **<http://127.0.0.1:8080/index.htm>** (use this link if the localhost link fails). Instructions for starting the J2EE Server can be found below.

**Note:** Starting the J2EE Server will cause a new window to be opened in which the server will run; do not close this window or you will shut down the server.

---

## Chapter 3: Working with PEERNET Reports Sample J2EE Solution

PEERNET Reports Sample J2EE Solution contains both a complete Java Web Application that is deployable to a Java Web Application Server, and an embeddable engine. The Java Web Application is built on top of the embeddable engine, and provides a jumpstart to using the technology. The Java Web Application is referred to as **PEERNET Reports Server**.

If you need to embed PEERNET Reports into an existing Java solution (such as a servlet, EJB, or application), you must use the PEERNET Reports Object Model API (which is a callable Java class library). For complete information, refer to the **PEERNET Reports Embedded Engine User Guide**.

### Using PEERNET Reports Server

PEERNET Reports Server is a Web Application that can be deployed to any Java Application Server running JDK 1.4 or greater. Note that PEERNET Reports Server can be deployed to a server running JDK 1.3, but some font and printing operations will not work as configured in PEERNET Reports Designer due to limitations or bugs in JDK 1.3.

You can manage PEERNET Reports Server on the Web Application's home page located at <http://<webserver>:<port>/PEERNETReports2/index.htm>. From this page you can activate the server, retrieve a list of the deployed projects, and also retrieve a list of the fonts available for use by any deployed project.

For the base installation, once you have started the J2EE Server, you can access the PEERNET Reports Server home page at <http://localhost:8080/PEERNETReports2/index.htm> or <http://127.0.0.1:8080/index.htm> (use this link if the localhost link fails). Note that starting the J2EE Server will cause a new window to be opened in which the server will run; do not close this window or you will shut down the server.

PEERNET Reports Server makes a request defined in a PEERNET Reports project available through a URL, or Universal Resource Locator (note that a URL must follow the industry standards for encoding and syntax for it to be a valid URL request). The request's result is available to any web browser or Web Application that references the URL. The URL you create must be of the following form:

<http://<webserver>:<port>/PEERNETReports2/PEERNETReportsServerRequestServlet?PROJECT=<project>&<REQUEST>=<request>>

where **PROJECT=<project>** provides the name of a project deployed to PEERNET Reports Server (for example, **PROJECT="demos.pnj"**) and **REQUEST=<request>** provides the name of a request defined in the project. Note that, if a request requires parameters, they are added to the end of the URL as **<parameter name>=<parameter value>**.

For the base installation, start the J2EE Server, then copy or type the following URL into your web browser:

<http://127.0.0.1:8080/PEERNETReportsServer2/PEERNETReportsServerRequestServlet?PROJECT=examples/labels/IndustryStandardLabels.pnj&REQUEST=JCPennyAsAdobePDF>

This request will respond with an Adobe PDF document. If you open the project (PNJ) file deployed to PEERNET Reports Server at **examples/labels/IndustryStandardLabels.pnj**, you will see

---

that server requests have been defined for this project. The configuration of the request determines the type of response returned by the URL.

## Printing reports and labels from a web page

PEERNET Reports Sample J2EE Solution allows reports or labels to be previewed and printed from web pages in several ways. For single-page reports or labels you can:

- embed an image of the report or label into a web page and print using the web browser
- link a report or label PDF to a web page and print using the Adobe Acrobat Reader
- use the PEERNET Reports Applet to automate the preview and printing operations

For multi-page reports or labels you can:

- link a report or label PDF to a web page and print using the Adobe Acrobat Reader
- use the PEERNET Reports Applet to automate the preview and printing operations

## Embedding report or label images into a web page

To embed an image into a web page, you must first define a request with a **ConvertTo** configured to use one of the PEERNET Image Writers. Then, embed an HTML image tag of the form:

```
<IMG src="http://<webserver>:<port>/PEERNETReportsServer2/
PEERNETReportsServerRequestServlet?PROJECT=<project>&REQUEST=<request>" WIDTH=100%
HEIGHT=100%>
```

You can also use a visual HTML editor to perform this task; set the source of the image to the URL, and use the editor to size the image display area.

## Linking report or label PDF documents to a web page

To embed a link to a report or label generated as an Adobe PDF document into a web page, you must first define a request with a **ConvertTo** configured to use the PEERNET Adobe PDF Writer. Then, embed an HTML link tag of the form:

```
<a href="http://<webserver>:<port>/PEERNETReportsServer2/
PEERNETReportsServerRequestServlet?PROJECT=<project>&REQUEST=<request>">Get PDF
Document</a>
```

You can also use a visual HTML editor to perform this task; highlight the text to be linked to the Adobe PDF Document, and set the source of the link to the URL.

## Working with the PEERNET Reports Applet

The PEERNET Reports Applet allows you to automate the preview and printing operations, and can also be used to automatically print reports or labels without first previewing them. As well, the PEERNET Reports Applet has a callable interface, so it can be used with JavaScript.

To use the PEERNET Reports Applet, you must first define a request with a **ConvertTo** configured to use the PEERNET Print Applet Writer. Then, embed an HTML applet tag of the form:

```
<APPLET NAME="PEERNETReportsApplet1" ARCHIVE=applet/PEERNETReportsApplet1.jar
CODE="com.peernet.PeernetReportApplet1.PeernetReportApplet1.class" width="473"
height="281">
<PARAM NAME="PREVIEW" VALUE="Yes">
PEERNET Reports Applet requires Sun Java Plugin 1.4 or better!
```

---

```
</APPLET>
```

```
<APPLET NAME="PEERNETReportsApplet2"  
ARCHIVE=http://<webserver>:<port>/PEERNETReportsServer2/applet/PEERNETReportsApplet  
1.jar  
CODE="com.peernet.PeernetReportApplet1.PeernetReportApplet1.class" WIDTH="16"  
HEIGHT="1">
```

```
    PEERNET Reports Applet requires Sun Java Plugin 1.4 or better!
```

```
</APPLET>
```

```
<APPLET NAME="PEERNETReportsApplet1" ARCHIVE=applet/PEERNETReportsApplet1.jar  
    CODE="com.peernet.PeernetReportApplet1.PeernetReportApplet1.class" width="473"  
height="281">
```

```
  <PARAM NAME="PREVIEW" VALUE="Yes">
```

```
  <PARAM NAME="URL"
```

```
VALUE="http://<webserver>:<port>/PEERNETReportsServer2/PEERNETReportsServerRequestS  
ervlet?PROJECT=<project>&REQUEST=<request>">
```

```
    PEERNET Reports Applet requires Sun Java Plugin 1.4 or better!
```

```
</APPLET>
```

```
<APPLET NAME="PEERNETReportsApplet1" ARCHIVE=applet/PEERNETReportsApplet1.jar  
    CODE="com.peernet.PeernetReportApplet1.PeernetReportApplet1.class" width="16"  
height="1">
```

```
  <PARAM NAME="URL"
```

```
VALUE="http://<webserver>:<port>/PEERNETReportsServer2/PEERNETReportsServerRequestS  
ervlet?PROJECT=<project>&REQUEST=<request>">
```

```
    PEERNET Reports Applet requires Sun Java Plugin 1.4 or better!
```

```
</APPLET>
```

You can also use a visual HTML editor to perform this task; insert the applet and set the properties as required.

You can then use JavaScript to perform the preview or printing operations. The code snippets below show you how to call the **printReport** or **previewReport** methods of the applet. If you set the **URL** parameter, the applet will automatically retrieve the content of the URL. If you set the parameter **PREVIEW** to **Yes**, the applet will preview the document and provide navigation and print buttons to the user; otherwise, the applet will automatically print the document with no user interaction.

```
function PrintDocument( showdialog )  
{  
  
    cmd =  
"http://<webserver>:<port>/PEERNETReportsServer2/PEERNETReportsServerRequestServlet  
?PROJECT=<project>&REQUEST=<request>";  
  
    document.PEERNETReportsApplet2.printReport( cmd, null, showdialog ) ;  
  
    status2 = document.PEERNETReportsApplet2.getStatus() ;  
  
    //    Wait for load of document to complete  
    while ( status2 & 1 )  
        status2 = document.PEERNETReportsApplet2.getStatus() ;  
  
    // Wait for document to be sent to the printer  
    while( status2 & 16)  
        status2 = document.PEERNETReportsApplet2.getStatus() ;  
  
    state = "No Status" ;
```

---

```
    if ( status2 & 1 )
        state = "Loading document from specified location" ;
    if ( status2 & 2 )
        state = "Finished Loading document from specified location" ;
    if ( status2 & 4 )
        state = "Failed to load document from specified location" ;

    if ( status2 & 16 )
        state = "Printing the document " ;
    if ( status2 & 32 )
        state = "Finished printing the document " ;
    if ( status2 & 64 )
        state = "Failed to print the document " ;

}

function LoadDocument()
{
    cmd =
    "http://<webserver>:<port>/PEERNETReportsServer2/PEERNETReportsServerRequestServlet
?PROJECT=<project>&REQUEST=<request>";

    document.PEERNETReportsApplet1.previewReport( cmd, null, false ) ;

    status2 = document.PEERNETReportsApplet1.getStatus() ;

    //    Wait for load of document to complete
    while ( status2 & 1 )
        status2 = document.PEERNETReportsApplet1.getStatus() ;

    state = "No Status" ;

    if ( status2 & 1 )
        state = "Loading document from specified location" ;
    if ( status2 & 2 )
        state = "Finished Loading document from specified location" ;
    if ( status2 & 4 )
        state = "Failed to load document from specified location" ;

}
```

---

## PEERNET Reports Applet

The PEERNET Reports Applet is a java applet that gets embedded in a web page to provide automated print functionality, file drop functionality or preview of reports or labels, and other functions that need to occur on the client's machine.

PEERNET Reports Applet is a form of AJAX support.

To embed the PEERNET Reports Applet into a web page you need to use the HTML APPLET tag of the form:

```
<APPLET ARCHIVE=http://<webserver>:<port>/.../PEERNETReportsApplet1.jar  
CODE="com.peernet.PeernetReportApplet1.PeernetReportApplet1.class">  
    PEERNET Reports Applet requires Sun Java Plugin 1.4 or better!  
</APPLET>
```

As with any applet you can specify the NAME, WIDTH, HEIGHT attributes. Refer to your HTML APPLET tag documentation for further information on these attributes.

The PEERNET Reports Applet allows for the following additional parameters to be passed to the applet:

Name	Description
DEBUG	<p>Turns on or off debugging information that is displayed in the java console.</p> <p>Values: True or False Yes or No</p> <p>Example: &lt;param name="DEBUG" value="true" /&gt;</p>
PREVIEW	<p>Turns on previewing of a report or label. When previewing is enabled, automatic printing is turned off and the users must press the print button on the print preview window in order to print the report or label.</p> <p>Values: True or False Yes or No</p> <p>&lt;param name="PREVIEW" value="true" /&gt;</p>

Name	Description
ZOOMTOFIT	<p>Turns on zoom to fit of a report or label into the preview window. This options is ignored with not previewing the report or label.</p> <p>Values: True or False Yes or No</p> <p>Example: &lt;param name=" ZOOMTOFIT " value="true" /&gt;</p>
SRC	<p>This parameter provides the URL where the applet will retrieve the information required for this operation.</p> <p>If the applet is printing or previewing a report or label this must be a URL that returns the PEERNET Print Applet language.</p> <p>If you are using the applet to copy information from the server to a port or file on the client machine, then the URL can return any type of data.</p> <p>Example: &lt;param name="SRC" value="http://&lt;webserver&gt;:&lt;port&gt;/PEERNETReportsServer2/PEERNETReportsServerRequestServlet?PROJECT=&lt;project&gt;&amp;REQUEST=&lt;request&gt;" /&gt;</p>
PORT	<p>This parameter is a port or file name on the client machine.</p> <p>When a PORT is specified, the applet performs a file copy operation from the source to the specified PORT. No interpretation of the data is made. This is a binary copy operation.</p> <p>Values: A valid port name installed on the client machine.</p> <p>Example: &lt;param name=" PORT " value="COM1:" /&gt; or &lt;param name=" PORT " value="C:/dropfile.txt" /&gt;</p>

Name	Description
PRINTER	<p>This parameter specifies the target printer when printing a report or label using the PEERNET Print Applet language.</p> <p>Values: A valid printer name installed on the client machine.</p> <p>Example: &lt;param name=" PRINTER " value="HP LaserJet 4050" /&gt;</p>
SHOWPRINTERDIALOG	<p>This parameter specifies the print selection dialog will appear when printing a report or label using the PEERNET Print Applet language.</p> <p>If the PRINTER parameter specifies a printer, this will be the initially selected printer otherwise, the default printer will be initially selected.</p> <p>Values: True or False Yes or No</p> <p>Example: &lt;param name="SHOWPRINTERDIALOG" value="true" /&gt;</p>
FontForMsgs	Font used to display text in dialog and message boxes displayed by the applet.
TitleForMsgs	Title to display in dialog and message boxes
PrintConfirmationMsg	<p>Confirmation message to display to confirm print action is to be performed.</p> <p>If no PrintConfirmationMsg is provided, then this confirmation is not preformed.</p>
PrintPickupMsg	<p>Pickup message to display after print action has completed.</p> <p>If no PrintPickupMsg is provided, then this pickup confirmation is not preformed.</p>
OkayButtonText	The text to be displayed on the Okay button. Default is OKAY.
CancelButtonText	The text to be displayed on the Cancel button. Default is Cancel.
RetryButtonText	The text to be displayed on the Retry button. Default is Retry.

---

In addition to the parameters, the PEERNET Report Applet may be accessed using JavaScript.

There are several APIs that may be call from JavaScript depending on your usage.

Method:

```
public synchronized void copyToPort( String url, String port )
```

Description:

The `url` can be any type of file. The `port` can be any valid port or file location that is valid for the client machine.

Purpose:

The purpose of the method is to allow a file to be transferred in binary mode from the server to the client.

For example, you may have a requirement where you produce native printer code (PostScript, Zebra, PCL, etc) and wish to send this directly to the printer. To accomplish this, you would copy from the `url` to the `port`

```
<yourappletname>.copyToPort( "http://...", "LPT1:")
```

For example, you may have a requirement where copy a file from your server to a file on the client's machine copy from the `url` to the `file`

```
<yourappletname>.copyToPort( "http://...", "c:/dropfile.pdf")
```

---

### Method:

```
public synchronized void printReport( String url, String printer, boolean  
bShowPrintDialog )
```

### Description:

The `url` can only PEERNET Print Applet content. The `printer` can be any valid printer installed on client machine or null (meaning use default printer). The `bShowPrintDialog` can be `true` or `false` and controls whether the print dialog is displayed during the print process.

### Purpose:

The purpose of the method is to allow a PEERNET Request whose convert to operation is setup to use the PEERNET Print Applet as its target to be printed on the client computer using the PEERNET Reports Applet. This operation can be performed with or without user interaction.

```
// Specifies the printer and print's without user interaction  
<yourappletname>.printReport( "http://...", "HP LasertJet 4050",  
false )  
  
// Does not specify a printer so the default printer will be used  
// and then specifies to prompt with the printer selection dialog  
<yourappletname>.printReport( "http://...", null, true )
```

---

### Method:

```
public synchronized void previewReport( String url, String printer, boolean  
bShowPrintDialog, boolean bZoomToFit )
```

### Description:

The `url` can only PEERNET Print Applet content. The `printer` can be any valid printer installed on client machine or null (meaning use default printer). The `bShowPrintDialog` can be `true` or `false` and controls whether the print dialog is displayed during the print process. The `bZoomToFit` can be `true` or `false` and controls whether the previewed report or label is zoomed to fit by default in the print preview window.

### Purpose:

The purpose of the method is to allow a PEERNET Request whose convert to operation is setup to use the PEERNET Print Applet as its target to be previewed on the client computer using the PEERNET Reports Applet. From the preview window the client can page through the pages of the report or label and/or initiate a print operation.

---

Method:

```
public long getStatus ()
```

Description:

This method is used to get the status of the current `printReport` or `previewReport` operation.

The return is a long contain a set of flags that indicate the status of the operation:

0 means no action is currently being performed.

(1 & `getStatus()`) means url is being downloaded from the server.

(2 & `getStatus()`) means url is has been downloaded from the server successfully.

(4 & `getStatus()`) means url is has been downloaded from the server unsuccessfully.

(16 & `getStatus()`) means the downloaded document is printing.

(32 & `getStatus()`) means the downloaded document has been printed successfully.

(64 & `getStatus()`) means the downloaded document has been printed unsuccessfully.

Purpose:

The purpose of the method is to allow the JavaScript method that calls `printReport` or `previewReport` method to know the status of the operation. It's mostly used for the `printReport` method.

---

## Examples:

```
function PrintDocument( showdialog )
{
    cmd =
"http://<webserver>:<port>/PEERNETReportsServer2/PEERNETReportsServerRequestServlet
?PROJECT=<project>&REQUEST=<request>";

    document.PEERNETReportsApplet2.printReport( cmd, null, showdialog ) ;

    status2 = document.PEERNETReportsApplet2.getStatus() ;

    //    Wait for load of document to complete
    while ( status2 & 1 )
        status2 = document.PEERNETReportsApplet2.getStatus() ;

    // Wait for document to be sent to the printer
    while( status2 & 16)
        status2 = document.PEERNETReportsApplet2.getStatus() ;

    state = "No Status" ;

    if ( status2 & 1 )
        state = "Loading document from specified location" ;
    if ( status2 & 2 )
        state = "Finished Loading document from specified location" ;
    if ( status2 & 4 )
        state = "Failed to load document from specified location" ;

    if ( status2 & 16 )
        state = "Printing the document " ;
    if ( status2 & 32 )
        state = "Finished printing the document " ;
    if ( status2 & 64 )
        state = "Failed to print the document " ;

}

function LoadDocument()
{
    cmd =
"http://<webserver>:<port>/PEERNETReportsServer2/PEERNETReportsServerRequestServlet
?PROJECT=<project>&REQUEST=<request>";

    document.PEERNETReportsApplet1.previewReport( cmd, null, false ) ;

    status2 = document.PEERNETReportsApplet1.getStatus() ;

    //    Wait for load of document to complete
    while ( status2 & 1 )
        status2 = document.PEERNETReportsApplet1.getStatus() ;

    state = "No Status" ;

    if ( status2 & 1 )
        state = "Loading document from specified location" ;
```

---

```
if ( status2 & 2 )
  state = "Finished Loading document from specified location" ;
if ( status2 & 4 )
  state = "Failed to load document from specified location" ;
}
```

---

Method:

```
public String getDefaultPrinter()
```

Description:

This method returns the default printer on the client's machine.

This method does not work on all browsers but is known to work on IE5+, Firefox, and Mozilla with Java 1.4+.

Purpose:

The purpose of the method is to allow JavaScript to know the name of the default printer.

Method:

```
public int getPrinterCount() {
```

Description:

This method returns the total number of installed printers on the client's machine.

This method does not work on all browsers but is known to work on IE5+, Firefox, and Mozilla with Java 1.4+.

Purpose:

The purpose of the method is to allow you know the total number of installed printers on the client's machine.

Method:

```
public String getPrinter( int index )
```

Description:

This method returns the name of an installed printer on the client's machine given a `index` between 0 and (`getPrinterCount()-1`).

This method does not work on all browsers but is known to work on IE5+, Firefox, and Mozilla with Java 1.4+.

Purpose:

The purpose of the method is to allow retrieve the names of all the installed printers on the client's machine.