

***Fiscal eposprint for Microsoft Visual Studio 2010
Sample Project Guide***

REVISION HISTORY

A	All	Preliminary.	15 th February 2013	P. Barnett
---	-----	--------------	--------------------------------	------------

Index

1	Overview	4
1.1	Description	4
1.2	List of Related Documents	4
1.3	Abbreviations.....	4
1.4	Restrictions of Use	4
2	Restaurant Sample Program.....	5
2.1	Intelligent Printer Setup	Errore. Il segnalibro non è definito.
2.2	Restaurant Web Site Upload	Errore. Il segnalibro non è definito.
2.3	Restaurant Visual Studio 2010 Visual Basic Import	5

1 Overview

1.1 Description

The aim of this manual is to provide information to software designers who wish to make use of the Epson fiscal eposprint for Microsoft Visual Studio 2010 sample project in order to connect via TCP/IP to Epson intelligent fiscal printers. The sample project aids the developer in a basic understanding of the fundamentals and can be thought of a template starting point for development.

1.2 List of Related Documents

- Fiscal ePOS-Print Solution Development Guide
- Fiscal Printer Intelligent Features Guide
- Epson_FP_Manuale Operatore
- Protocollo di Comunicazione
- Intelligent Printer Printing System ePOS-Print Technical Data

1.3 Abbreviations

- FP: Fiscal Printer
- VB: Visual Basic

1.4 Restrictions of Use

- The sample program is for evaluation purposes only and is not intended to be used as a finished application in retail outlets.
- No part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Seiko Epson Corporation.
- The contents of this document are subject to change without notice. Please contact us for the latest information.
- While every precaution has taken in the preparation of this document, Seiko Epson Corporation assumes no responsibility for errors or omissions.
- Neither is any liability assumed for damages resulting from the use of the information contained herein.
- Neither Seiko Epson Corporation nor its affiliates shall be liable to the purchaser of this product or third parties for damages, losses, costs, or expenses incurred by the purchaser or third parties as a result of: accident, misuse, or abuse of this product or unauthorized modifications, repairs, or alterations to this product, or (excluding the U.S.) failure to strictly comply with Seiko Epson Corporation's operating and maintenance instructions.
- Seiko Epson Corporation shall not be liable against any damages or problems arising from the use of any options or any consumable products other than those designated as Original EPSON Products or EPSON Approved Products by Seiko Epson Corporation.

2 Restaurant Sample Program

The restaurant sample program can be used to print sample fiscal receipts (scontrini) and invoices (fatture). Simply select the items and watch the list grow and when ready choose either to print a fiscal receipt or an invoice. The intelligent fiscal printer responses can be checked under "STATO STAMPANTE:". The list is then cleared ready for the next sample. Invoice numbers are also incremented.

2.1 Restaurant Visual Studio 2010 Visual Basic Import

Item	Description
Tested environment	Microsoft Visual Studio 2010
Required item	-
Sample program file name	fiscalprint_vb2010_restaurant-demo.zip
Source editing	Open the following file: fiscalprint_vb2010_restaurant-demo\ WindowsApplication2\restaurant_form.vb Edit the IP address of the intelligent fiscal printer by modifying the following line: <code>Private host As String = ""</code> The default IP address is "192.168.0.2".

1. Check the connectivity between the computer and intelligent fiscal printer using ping for example.
2. Uncompress the sample program into any folder.
3. Open the Restaurant_demo.vbproj sample program solution file using Visual Studio 2010.

2.2 Data Structure

This demo application is composed of two parts:

- A library (fiscalprint_lib)
- An application which makes use of the library (WindowsApplication2)

2.3 Library Functions

The library exposes methods that the application uses to send data to the printer.

2.3.1 setDataToSend(ByRef data As String)

This method is used to set the XML content that will be sent to the printer. The data string argument should contain all the receipt/invoice XML lines except for the following tags which are automatically inserted by the library:

- ?xml
- s:Envelope
- s:Body

default:

```
<printerCommand>  
<Printer Num="1" />  
<queryPrinterStatus />  
</printerCommand>
```

2.3.2 setTimeout(ByVal I_tout As Integer)

This method is used to set the transaction timeout (in 10mS units). Once this timeout expires, the request is deemed unanswered and fails. This timer should be lengthened for very long receipts.

default: 500 (5 seconds)

2.3.3 setDevId(ByVal I_devid As String)

This method is used to set the printer identifier or device id and instructs the fiscal printer that receives the XML to either print locally or to a secondary printer.

default: "local_printer"

2.3.4 setHost(ByVal I_host As String)

This method is used to set the IP address of the printer that will receive the XML request.

default: "192.168.0.2"

2.3.5 setDOM(ByRef mydom As MSXML2.DOMDocument60)

This method must be called to set the DOM that will subsequently contain the transaction response from the printer.

2.3.6 Send()

This method instructs the library to perform the transaction (send the XMLHTTP request to the printer).

2.4 Variables

The following variable needs to be taken into account:

- Private InvoiceNumber As Integer = 3000

The intelligent fiscal printer needs to be programmed with an invoice number (NUMERO FATTURA) ≤ 3000 . If the printer invoice number is set to a higher number, printing will fail and a "printer error" message will be visible in the printer response window. Alternatively, set this variable to a value $\geq$ the printer setting. The printer automatically increments this number after having printed an invoice – the following line maintains the synchronisation:

```
InvoiceNumber = InvoiceNumber + 1
```