ZBI-Developer™

User Guide

48789L-002

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.

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

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

Limitation of Liability In no event shall Zebra Technologies 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 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.

Part Number: 48789L-002

Contents

Notes • ____

____

____

____

____

____

____

____

____

About This Document

This section provides you with contact information, document structure and organization, and additional reference documents.

Contents

Who Should Use This Document

This User Guide helps you quickly develop, test, and distribute ZBI programs using **ZBI-Developer™**.

How This Document Is Organized

The User Guide is set up as follows:

Section	Description
Introduction - Welcome to ZBI-Developer	An introduction to **ZBI-Developer**
Installation	How to install the program
The Interface	Describes the ZBI-Developer interface
Configuration of ZBI-Developer	How to set up and customize software to your preferences
ZBI-Developer Tutorial	How to create, test, and distribute a new ZBI programs using the ZBI-Developer environment
Encryption of ZBI Programs	How to optionally encrypt programs before distributing them
Encryption of ZBI Programs	How to contact technical support

Contacts

Technical Support via the Internet is available 24 hours per day, 365 days per year.

Web Site: www.zebra.com
E-mail Back Technical Library:
E-mail address: emb@zebra.com
Subject line: Emaillist
Self Service Knowledge Base: www.zebra.com/knowledgebase
Online Case Registration: www.zebra.com/techrequest

Which Department Do You Need?	The Americas	Europe, Africa, Middle East, India	Asia Pacific
Regional Headquarters	Zebra Technologies International, LLC 333 Corporate Woods Parkway Vernon Hills, IL 60061-3109 U.S.A. T: +1 847 793 2600 Toll-free +1 800 423 0422 F: +1 847 913 8766	Zebra Technologies Europe Limited Dukes Meadow Millboard Road Bourne End Buckinghamshire, SL8 5XF United Kingdom T: +44 (0) 1628 556000 F: +44 (0) 1628 556001	Zebra Technologies Asia Pacific Pte. Ltd. 120 Robinson Road #06-01 Parakou Building Singapore 068913 T: + 65 6858 0722 F: +65 6885 0838
Technical Support For questions on the operation of Zebra equipment and software, please call your distributor. For additional assistance, contact us. *Please have your model and serial numbers available.*	T: +1 877 ASK ZEBRA (275 9327) F: +1 847 913 2578 Hardware: ts1@zebra.com Software: ts3@zebra.com *Kiosk printers:* T: +1 866 322 5202 E: kiosksupport@zebra.com	T: +44 (0) 1628 556039 F: +44 (0) 1628 556003 E: Tseurope@zebra.com	T: +65 6858 0722 F: +65 6885 0838 E: *China:* tschina@zebra.com *All other areas:* tsasiapacific@zebra.com
Repair Service Department For back-to-base service and repair.	T: +1 877 ASK ZEBRA (275 9327) F: +1 847 821 1797 E: repair@zebra.com To request a repair in the U.S., go to www.zebra.com/repair.	T: +44 (0) 1772 693069 F: +44 (0) 1772 693046 New requests: ukrma@zebra.com Status updates: repairupdate@zebra.com	T: +65 6858 0722 F: +65 6885 0838 E: *China:* tschina@zebra.com *All other areas:* tsasiapacific@zebra.com
Technical Training Department For Zebra product training courses.	T: +1 847 793 6868 T: +1 847 793 6864 F: +1 847 913 2578 E: ttamerica@zebra.com	T: +44 (0) 1628 556000 F: +44 (0) 1628 556001 E: Eurtraining@zebra.com	T: + 65 6858 0722 F: +65 6885 0838 E: *China:* tschina@zebra.com *All other areas:* tsasiapacific@zebra.com
Inquiry Department For product literature and distributor and dealer information.	T: +1 877 ASK ZEBRA (275 9327) E: inquiry4@zebra.com	T: +44 (0) 1628 556037 F: +44 (0) 1628 556005 E: mseurope@zebra.com	E: *China:* GCmarketing@zebra.com *All other areas:* APACChannelmarketing@zebra.com
Customer Service Department (US) ***Internal Sales Department (UK)*** For printers, parts, media, and ribbon, please call your distributor or contact us.	T: +1 877 ASK ZEBRA (275 9327) E: clientcare@zebra.com	T: +44 (0) 1628 556032 F: +44 (0) 1628 556001 E: cseurope@zebra.com	T: +65 6858 0722 F: +65 6885 0836 E: *China*: order-csr@zebra.com *All other areas:* csasiapacific@zebra.com

Key:
T: Telephone
F: Facsimile
E: E-mail

Document Conventions

The following conventions are used throughout this document to convey certain information.

Alternate Color (online only) Cross-references contain hot links to other sections in this guide. If you are viewing this guide online in .pdf format, you can click the cross-reference (blue text) to jump directly to its location.

Command Line Examples Command line examples appear in `Courier New` font. For example, type `ZTools` to get to the Post-Install scripts in the `bin` directory.

Files and Directories File names and directories appear in `Courier New` font. For example, the `Zebra<version number>.tar` file and the `/root` directory.

Notes and Examples

Note • Indicates neutral or positive information that emphasizes or supplements important points of the main text.

Example • Provides an example, often a scenario, to better clarify a section of text.

Related Documents

The following documents might be helpful references:

Applicable Zebra Printer User Guide
ZPL II® Programming Guide
ZebraNet® 10/100 Print Server User Guide
ZebraNet® Wireless Print Server User Guide

Notes • ____

____

____

____

____

____

____

____

____

____

ZBI-Developer

Introduction - Welcome to ZBI-Developer

ZBI-Developer is an Integrated Development Environment (IDE), designed to assist in the creation, testing and distribution of programs written for use in the Zebra Basic Interpreter™ (ZBI™).

ZBI-Developer offers several ease-of-use features:

- Project File Control – Keeps related files in a single Project folder
- Pop-up Help – displays ZBI command syntax and examples
- Virtual Printers – allows ZBI programs to be run on the PC, with no need for a printer
- Multiple Perspectives – displays different display modes for the software:
  - ZBI Perspective – used for writing ZBI programs
  - Debug Perspective – used for testing and checking programs
- Multiple Views – displays different information screens:
  - Navigator View – used to manage Projects and files in projects
  - Printer View – used to manage physical and virtual printers
  - Debug View – used to display the name of the program being tested
  - Breakpoints View – used to view the list of breakpoints and jump to each point
  - Variable View – used to view the current value of variables in the program
  - Problems View – displays Warnings and Errors detected in the ZBI program
  - Ports View – displays data as it flows through ports used in the program
- Program Distribution – programs can be easily sent to a printer
- Program Encryption – programs can be encrypted in order to protect proprietary data
- File Drag and Drop – files can be dragged into a project from the Windows® Explorer environment
- File Import – files can be imported from printers into Projects in ZBI-Developer
- File Compare – files can be compared to earlier versions of the same file or to other files

System Requirements

These are the minimum requirements needed to run ZBI-Developer™

- Pentium®-based computer with at least 64 MB of RAM
- One of the 32-bit Windows® operating system:
  - Windows 2000®
  - Windows XP®
  - Windows Vista®
  - Windows Server 2003®
- Hard disk with 80 MB of free disk space
- Administrator rights to the local computer during installation

Printers, ZBI Keys & ZBI Versions

Information about ZBI 1.x and ZBI 2.x:

ZBI versions 1.0 through 1.5:

ZBI 1.x was available on printers with X.10 or higher firmware (such as V48.10.x). To determine if the printer supports ZBI version 1, check the firmware version loaded on the printer. This can be determined by the absence of a “Z” in the firmware version number (for example, firmware V60.13.0.12 supports ZBI version 1, while V60.13.0.12Z does not). The following printers support the ZBI 1.x firmware:

- *LP/TLP 284x-Z* and *384x-Z*
- *S300/S400/S500/S600*
- Z4000/Z6000
- Z4M/Z6M
- Z4Mplus/Z6Mplus
- 105*SL*
- *PAX*3
- *Xi*II
- *Xi*III

ZBI-Developer can be used to create programs for use on printers that support ZBI version 1.x., however, the features that are only available in ZBI v2.x cannot be used with printers running ZBI v1.x. For example, “on-printer” debugging advanced file encryption and commands added in ZBI 2 are not supported in printers running ZBI 1.x. If you do not have a printer that meets this requirement, contact your reseller.

> **Note •** Support for ZBI versions 1.0 through 1.5 is limited to syntax checking only. On-printer debugging is not supported for ZBI versions 1.0 through 1.5.

ZBI versions 2.0 and higher:

Printers with firmware versions X.16 or later (for example, V60.16.x and V53.16.x) can support ZBI version 2.0 and later. The following printers support the ZBI 2.x firmware:

- *Xi*III*Plus*
- Z4Mplus/Z6Mplus
- 105*SL*
- S4M
- *PAX*4
- ZM400/ZM600
- HC100
- G-Series
- Xi4

These printers can be either ZBI-Ready or ZBI-Enabled, depending on whether or not a ZBI Key file has been loaded on the printer. ZBI Keys can be loaded onto printers during manufacturing or later purchased at www.zebrasoftware.com. A Downloader Utility/ZBI Key Manager software utility is available to assist in the task of sending ZBI Keys to printers.

The ZBI.nrd file is required to be present on the printer for ZBI 2.0 to be enabled. The ZBI Key is stored on the printer's E: memory location with the name ZBI.nrd. The file is persistent. It cannot be deleted even if the printer's memory is initialized. For example, if the `^JB` command is used to initialize the location, the ZBI Key file will not be deleted.

When a printer is ZBI-Ready but not ZBI-Enabled, the firmware version will display a "Z" at the end of the version string (for example, V60.16.0Z). Additionally, the printer's configuration label will show that the printer is not ZBI-Enabled.

When a printer is ZBI-Enabled, the firmware version will not display a "Z" at the end of the version string (for example, V60.16.0). Additionally, the printer's configuration label will show that the printer is ZBI-Enabled.

Note • Each single ZBI Key can only be used once. When multiple printers are to be ZBI-Enabled, multiple Keys will be needed. The ZBI Key cannot be retrieved from printer to a host system.

Supported Print Servers

These are the supported ZebraNet print servers:

- ZebraNet 10/100 Print Server (firmware V1.1.6 or later required to support on-printer debugging)
- ZebraNet Wireless Print Server (V60.16.x, V53.16.x or later firmware required)
- ZebraNet Wireless Plus Print Server (V60.16.x, V53.16.x or later firmware required)

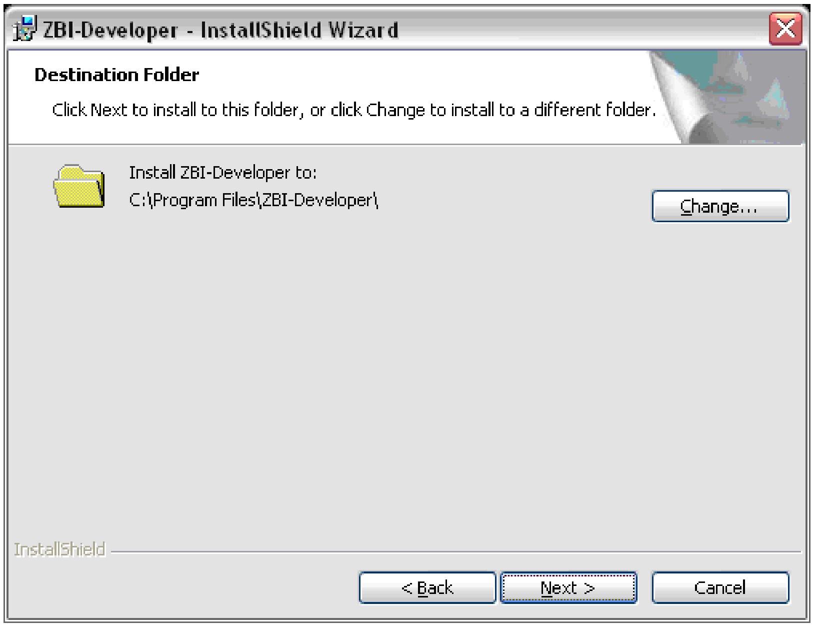
Installation

ZBI-Developer can be installed from a downloaded copy or from a CD-ROM containing the software. Before starting the ZBI-Developer installation, close all other software applications. This ensures a complete installation of all of the program files.

The complete installation package can be downloaded from www.zebra.com

> **Note •** If you are using Windows 2000, Windows XP, Windows Vista or Windows Server 2003, you must log on with Administrator rights to perform the installation.

During the installation, follow the user prompts to control the location in which to install the software and other options. If you change the default directory location for the software installation, be sure to note the new location. This information will be helpful to have for backup and archival purposes.



ZBI-Developer - InstallShield Wizard

Destination Folder

Click Next to install to this folder, or click Change to install to a different folder.

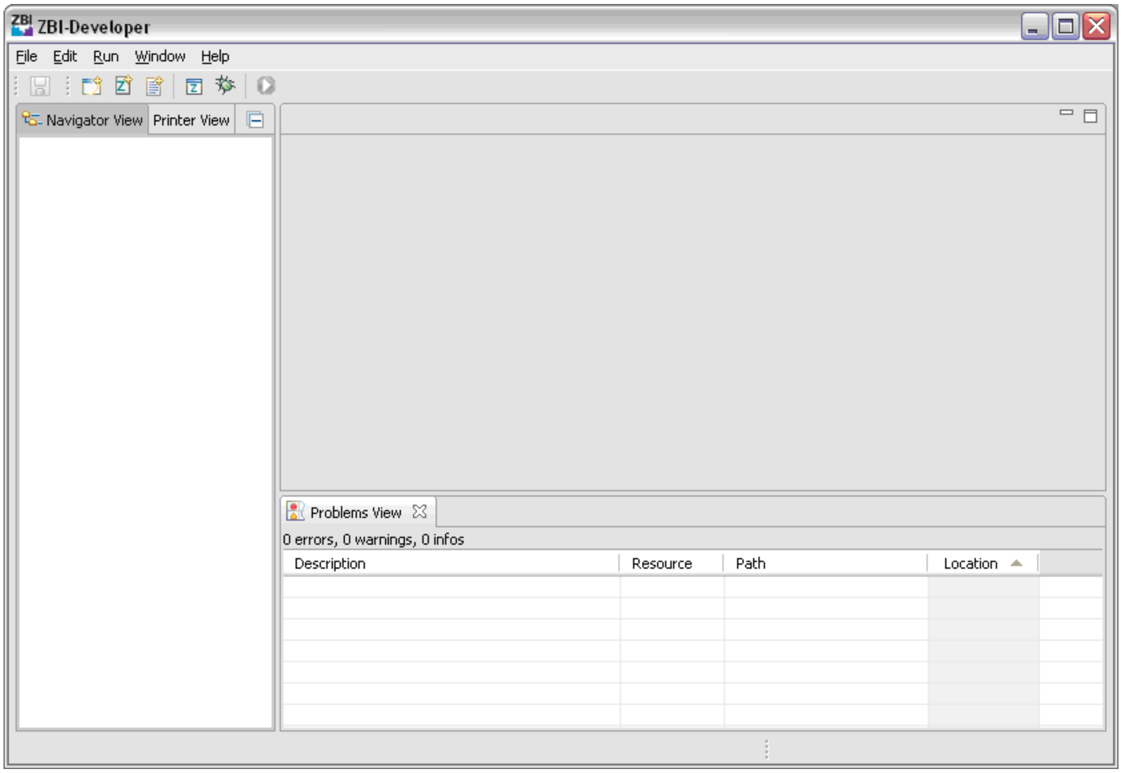
Install ZBI-Developer to:
C:\Program Files\ZBI-Developer\

Once the installation is complete, click **Finish** to complete.

Starting ZBI-Developer

ZBI-Developer can be started from the Windows Start menu – **Start > Programs > ZBI-Developer** or from any shortcuts created for the ZbiDeveloper.exe file.

When the program is run for the first time, the default interface, or workspace, will be displayed.



0 errors, 0 warnings, 0 infos

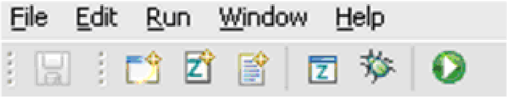
The Interface

The ZBI-Developer interface is designed to give the user quick access to the key features needed to develop ZBI programs. The interface is divided into various visual elements; these include the Menu bar, Toolbars, Perspectives, Views, and Editors that make up the interface.

- **Menu and Toolbars** – display quick access to the various functions and features of ZBI-Developer
- **Projects** – used to manage the files associated with creating a ZBI program
- **Perspectives** – display different use modes for the software:
  - ZBI Perspective for writing ZBI programs
  - Debug Perspective for testing and checking programs.
- **Views** – display different information screens:
  - Navigator View – used to manage Projects and files in projects
  - Printer View – used to manage physical and virtual printers
  - Debug View – used to display the name of the program being tested
  - Breakpoints View – used to view the list of breakpoints and jump to each point
  - Variable View – used to view the current value of variables in the program
  - Problems View – displays Warnings and Errors detected in the ZBI program
  - Ports View – displays data as it flows through ports (serial, parallel, IP) used in a ZBI program
- **Editors** – Used to create and alter files. The appropriate editor is displayed for the selected file type.

Menu and Toolbars

The Menu and Toolbars are located at the top of the screen. They provide access to the File, Edit, Run, Window, and Help submenus and commonly used program features.



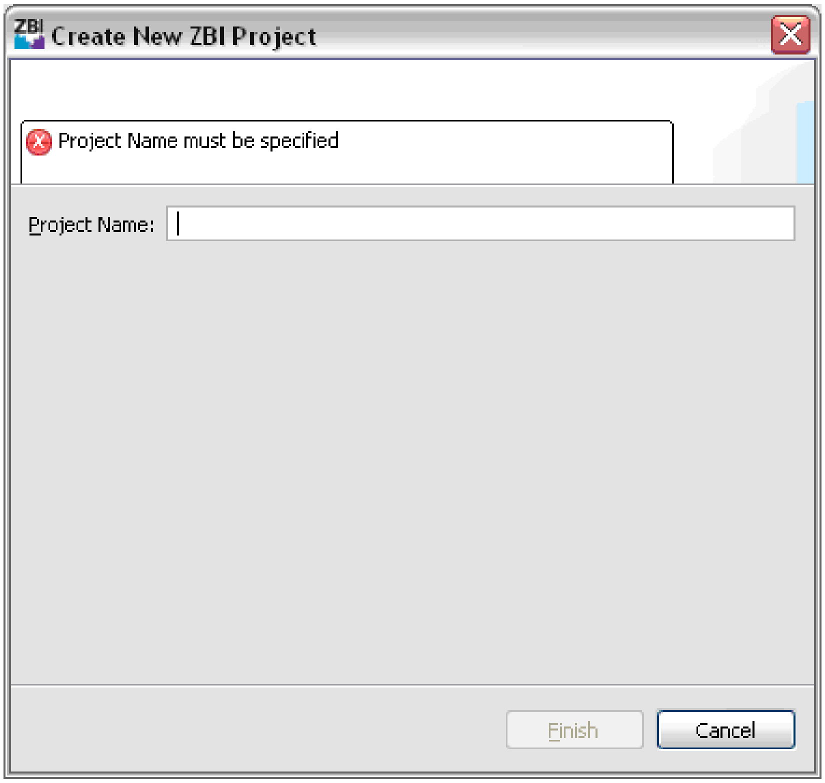
File Menu

The **File** menu enables you to create, save, close, print, import, and export resources, and exit ZBI-Developer.

New Project

To create a new Project, select **File** > **New** > **New Project.**

The New ZBI Project Wizard will display, with prompts for creating a new Project. Use the Wizard prompts to fill in the Project name (required). Click **Finish** to complete the Wizard.

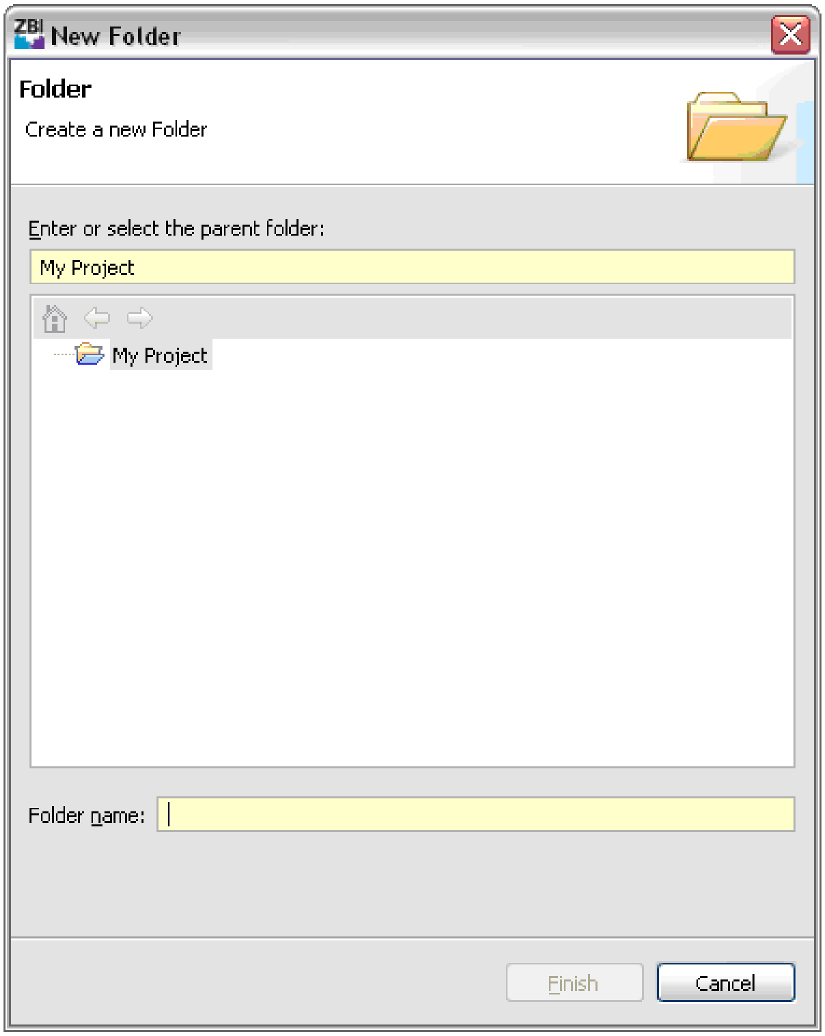

Create New ZBI Project

Project Name must be specified

New Folder

To create a new Folder within a Project, click on a Project in the Navigator View to select it and click **File > New > New Folder**.

The New Folder Wizard will display, with prompts for creating the New Folder. Use the Wizards prompt to fill in the New Folder name. Click **Finish** to complete the Wizard.



New Folder

Folder

Create a new Folder

Enter or select the parent folder: My Project

Folder name:

New ZBI Program

To create a new ZBI Program, select **File > New > New ZBI Program.** A new, Untitled ZBI program will open and display in the ZBI-Developer workspace. The program will not appear in the Project Folders until it is named and saved. To Save and Name a program, type or paste content into the program and click the **File > Save** menu selection.

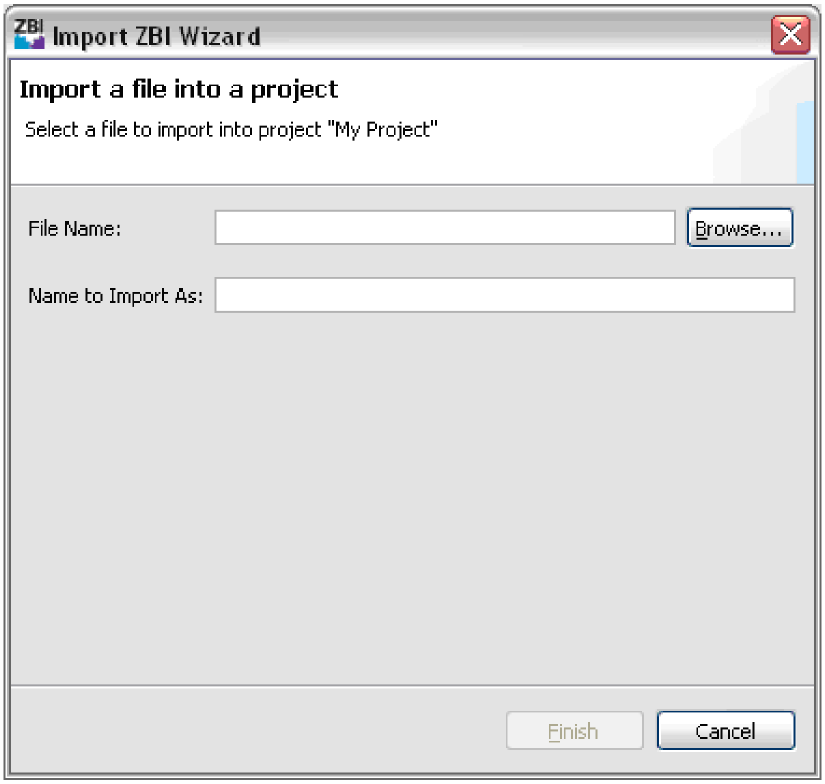
New Untitled File

To create a new Untitled File, select **File > New > New Untitled File.** Untitled Files are usually created to store data important to the Project, such as data or input streams that the ZBI program will use during execution. Untitled Files can also be used to create Notes or documentation for the Project. Clicking the New Untitled File icon has the same effect. A new, Untitled File will open and display in the ZBI-Developer workspace. The File will not appear in the Project Folders until it is named and saved. To Save and Name a program, type or paste content into the program and click the **File > Save** menu item.

Import a File

To Import a file from the computer’s hard drive into a Project, select **File > Import….** Files cannot be imported unless they are being directed into a Project.

When **Import…** is selected, the Import Wizard will be displayed. Use the **Browse** button to select the file to import. Use the **Name to Import As** textbox to alter the name as needed. Files with .zbi extensions will be imported as .zbi files. Other file types will retain their original file extension. If a file does not have an extension, it will be given a .txt extension.



Import ZBI Wizard

Import a file into a project

Select a file to import into project "My Project"

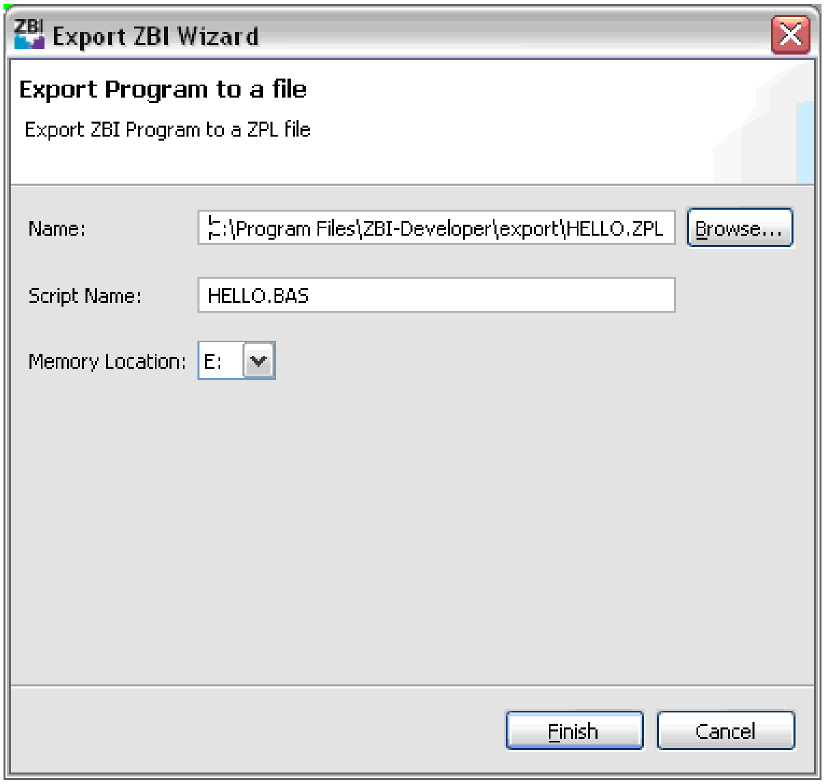
File Name:

Name to Import As:

Exporting Files

To Export ZBI Programs and Encryption Key Files from the ZBI-Developer environment to the computer’s hard drive, click on a file and then select **File > Export…**.

Based on the choices entered in the Export Wizard, the name of the file and the printer memory location to store it in will be written automatically into the Export file.



Export Program to a file

Export ZBI Program to a ZPL file

Name: E:\Program Files\ZBI-Developer\export\HELLO.ZPL

Script Name: HELLO.BAS

Memory Location: E:

The purpose of the Export ZBI Program feature is to create a file that can be distributed to one or more printers. During the Export process, ZBI programs will be exported in a format for use on a printer. The program will be MIME encoded, with all REM statements removed from the program.

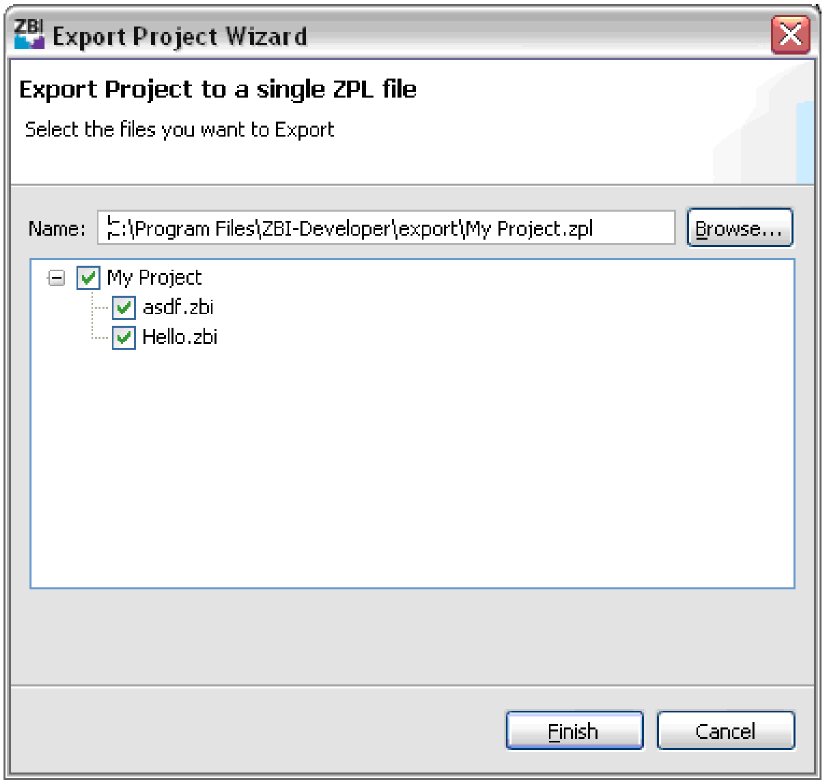
> **Note:** In order for an encrypted program to be executable, the matching encryption key must also be present in the printer.

If the ZBI program has been Encrypted, then the Exported file will be Encrypted and MIME encoded. This allows the programmer to create a file that can be emailed to users without revealing proprietary information.

Exporting Projects

To Export a ZBI Project to a single file on the computers hard drive, click on a Project folder and then select **File** > **Export…**

Based on the choices entered in the Export Wizard, the files in the project will be written to a single file on the hard drive.



Export Project Wizard

Export Project to a single ZPL file

Select the files you want to Export

Name: C:\Program Files\ZBI-Developer\export\My Project.zpl

☑ My Project
☑ asdf.zbi
☑ Hello.zbi

The purpose of the Export feature is to create a file that can be distributed to one or more printers. During the Export process, ZBI programs will be exported in a format for use on a printer. The program will be MIME encoded, with all REM statements removed from the program.

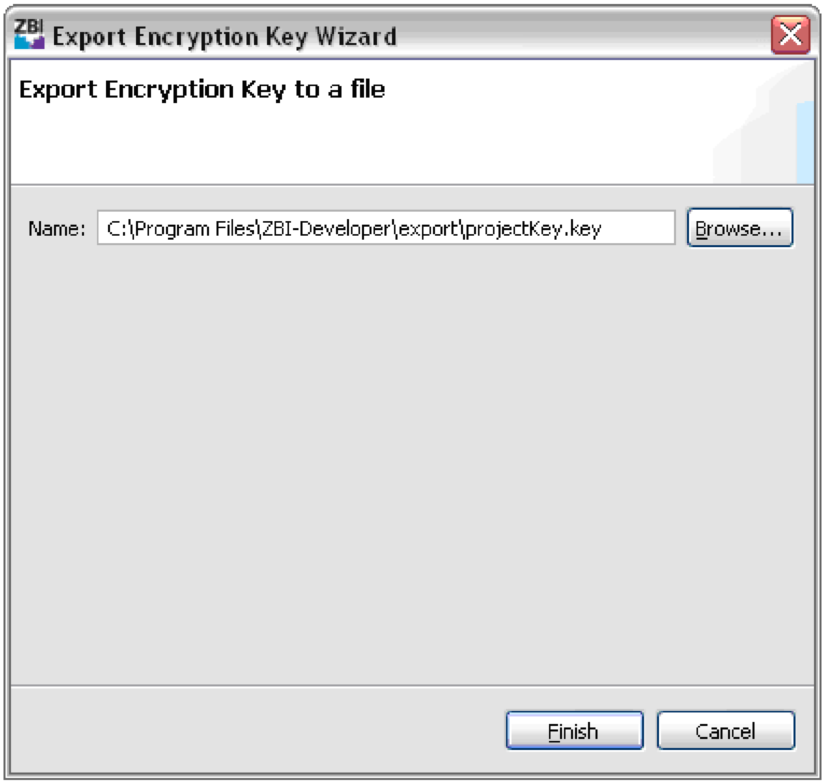
If the ZBI program has been Encrypted, then the Exported file will be Encrypted and MIME encoded. This allows the programmer to create a file that can be emailed to users without revealing proprietary information. To use this feature, it is necessary to first use the **Export Encryption Key** feature

Exporting Encryption Keys

The Encryption Key can be Exported and then sent to the printer. This feature allows for the following scenario – a Developer can create a ZBI program and then encrypt it. The Encryption Key can then be downloaded to the printer – thus allowing later updates of the program to also be downloaded to the printer. In this way, the Developer can protect their source code, while also allowing for program updates.

To Export a ZBI Encryption Key to a single file on the computers hard drive, click on a Encryption Key file and then select **File > Export…**

Based on the choices entered in the Export Wizard, the Encryption Key will be written to a single file on the hard drive.



Export Encryption Key Wizard

Export Encryption Key to a file

Name: C:\Program Files\ZBI-Developer\export\projectKey.key

The exported file can now be sent to printers as needed.

Note • The Encryption Key must be installed on the printer by the programmer so as to maintain control over the ability to put the Encrypted programs on printers. The Key should not be distributed to end users.

Close

The **Close** menu selection will close the currently selected file.

Close All

The **Close All** menu selection closes all currently open files.

Save

The **Save** menu selection will Save the currently selected file. If the file has a .zbi extension, it will be saved as a ZBI program. If the file is an Untitled File, it will be necessary to give the file an extension when saving it.

In all cases, if multiple Projects exist, it is necessary to specify which Project to save the file in.

The program will display “Invalid File Name” if no name is entered or if the “.” character is used but no extension is entered.

> **Note •** ZBI-Developer will attempt to make use of the file extension associations currently in use in the operating system. This means that if a file is saved as a .doc file, ZBI-Developer will attempt to call the file viewer associated with that file type.

Save As

The **Save As** menu selection will Save the currently selected file with the opportunity to rename the file, change the file’s extension, and alter the Project directory the file is stored in.

If the file has a .zbi extension, it will be saved as a ZBI program. If the file is an Untitled File, it will be necessary to give the file an extension when saving it. The program will display “Invalid File Name” if no name is entered or if the “.” (period) character is used, but no extension is entered.

In all cases, if multiple Projects exist, it is necessary to specify which Project to save the file in.

> **Note •** ZBI-Developer will attempt to make use of the file extension associations currently in use in the operating system. This means that if a file is saved as a .doc file, ZBI-Developer will attempt to call the file viewer associated with that file type.

Save All

The **Save All** menu selection will close all currently open files. If the file has a .zbi extension, it will be saved as a ZBI program. If the file is an Untitled File, it will be necessary to give the file an extension when saving it. The program will display “Invalid File Name” if no name is entered or if the “.” (period) character is used but no extension is entered.

In all cases, if multiple Projects exist, it is necessary to specify which Project to save previously unsaved files in.

> **Note •** ZBI-Developer will attempt to make use of the file extension associations currently in use in the operating system. This means that if a file is saved as a .doc file, ZBI-Developer will later attempt to call the file viewer associated with that file type.

Revert

Choosing **Revert** will alter the content of the current Editor back to the content of the last saved file. The **Revert** menu choice is disabled if the editor does not contain any unsaved changes.

> **Note •** Do not confuse **Revert** with **Undo**. The **Undo** feature reverses you most recent editing action, while **Revert** returns the contents of the editing window to the saved contents of the file.

Rename

Rename allows the user to alter the name of a previously created Project or a file within a Project. A Folder or File must be selected for the **Rename** menu selection to be active. Only one Project or File can be renamed at a time.

Print

Print allows the currently selected and open file to be printed.

Exit

Selecting **Exit** closes all open files. The program will confirm if recent unsaved changes should be saved.

Edit Menu

The **Edit** menu helps you alter, search, and manage files in the ZBI-Developer editor environments.

Undo

This command will reverse the most recent edit.

Redo

The **Redo** feature will re-apply the last edit that was reversed by the most recent **Undo** action.

Cut

Removes the selected text and places it on the clipboard.

Copy

Copies the selected text to the clipboard.

Paste

Places the text currently on the clipboard in the open document at current cursor location.

Delete

Removes the currently selected item.

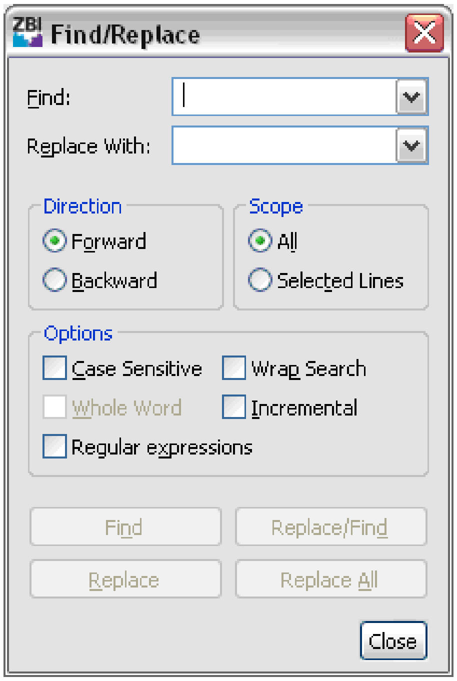
> **Note •** The **Delete** feature can remove selected text, or an object such as a file or an entire Project.

Select All

Selects all objects or text in the current view or editor.

Find/Replace

The **Find** feature is used to search the current editor for an expression. The optional **Replace** feature allows the user to replace an expression with another expression.



Find/Replace

Find:

Replace With:

Direction
● Forward
○ Backward

Scope
● All
○ Selected Lines

Options
□ Case Sensitive □ Wrap Search
□ Whole Word □ Incremental
□ Regular expressions

Find Next

Allows the user to search for the next occurrence of the selected text or the next occurrence of the most recently found expression.

Find Previous

Allows the user to search for the previous occurrence of the selected text or the previous occurrence of the most recently found expression.

Word Completion

This feature is offered when editing text files. When selected, it will attempt to complete the current word.

Set Encoding

This feature is offered when viewing text and other files that are not ZBI programs. The following encoding types can be set:

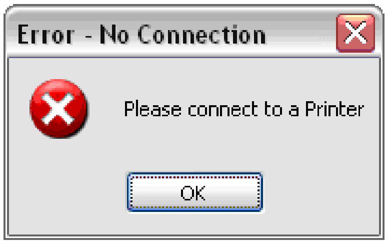
- Codepage1252 (Cp1252)
- US-ASCII
- UTF-16
- UTF-16BE
- UTF-16LE
- UTF-8
- ISO-8859-1

To set or change the encoding type, right-click over a file and open the Properties dialog.

Run Menu

Run\Debug ZBI Application

This feature will cause ZBI-Developer to execute the selected ZBI program on the Connected printer. If no printer connection has been created, the program will display an Error dialog stating “No Connection. Please connect to a Printer”.



Error - No Connection

Please connect to a Printer

Window Menu

The **Window > Show Perspective** and **> Show View** menus are used to change the current View, Perspective, and access the **Preferences** dialog.

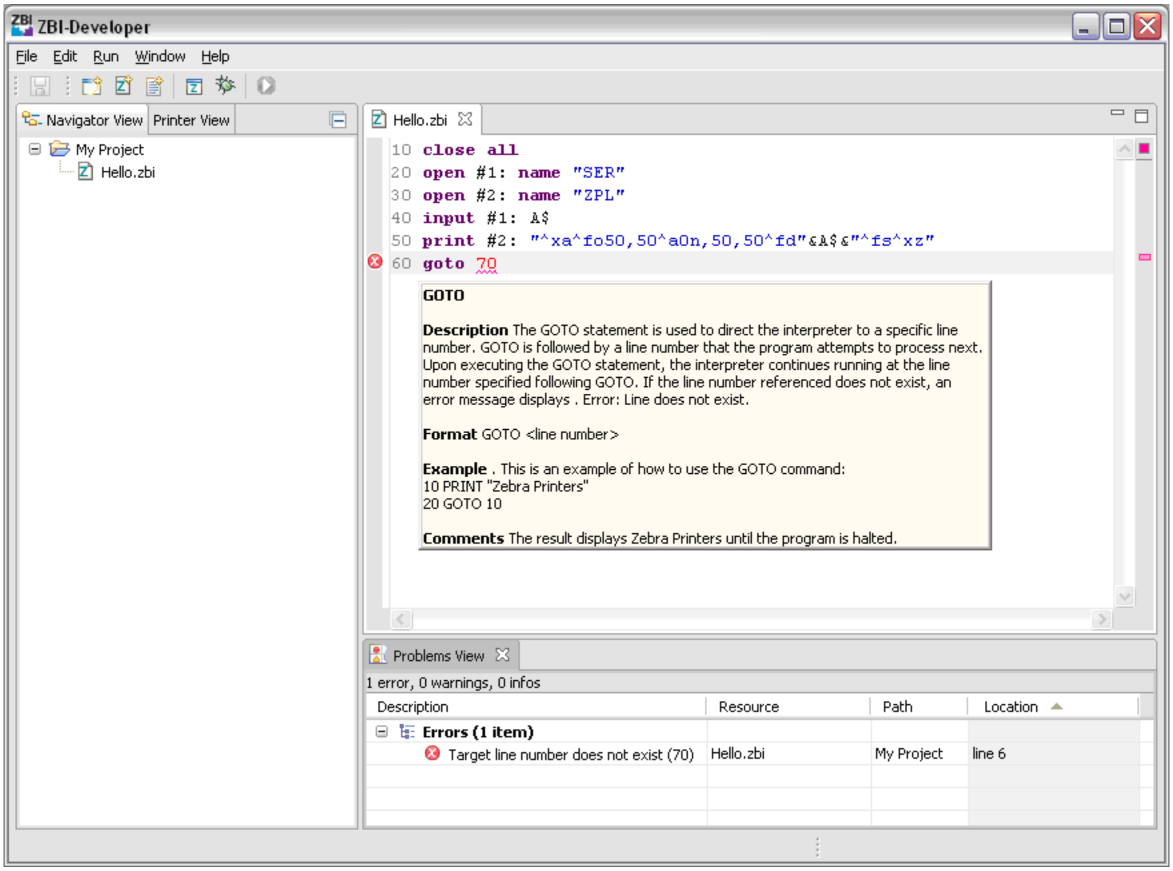
A Perspective defines the set and arrangement of **Views** and dialogs shown in the **ZBI-Developer** workspace. Within the window, each **Perspective** shares the same set of editors.

Each **Perspective** provides a set of features and functions aimed at accomplishing a specific task. For example, the **ZBI Perspective** combines **Views** that you would commonly use while editing ZBI project source files. The **Debug Perspective** contains the **Views** that you would use while debugging programs. As you work in the program, you will probably switch the **Perspective** frequently.

There are two Perspectives in the program, **ZBI** and **Debug.**

ZBI Perspective

The **ZBI Perspective** is used to create ZBI Projects and the actual ZBI program. By using the features of **Navigator View, Printer View, Problems View and Editor View**, the programmer will be able to quickly create a ZBI program.



```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 70
```

GOTO

Description The GOTO statement is used to direct the interpreter to a specific line number. GOTO is followed by a line number that the program attempts to process next. Upon executing the GOTO statement, the interpreter continues running at the line number specified following GOTO. If the line number referenced does not exist, an error message displays . Error: Line does not exist.

Format GOTO <line number>

Example . This is an example of how to use the GOTO command:
10 PRINT "Zebra Printers"
20 GOTO 10

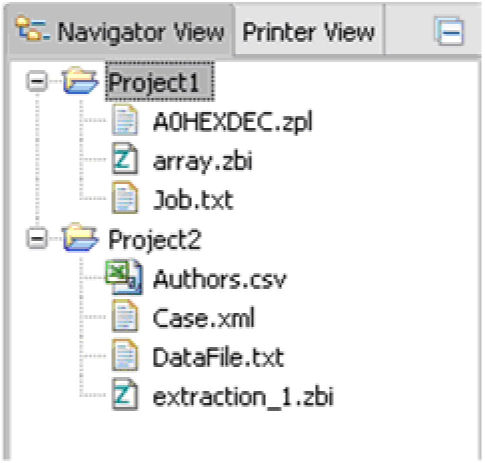
Comments The result displays Zebra Printers until the program is halted.

1 error, 0 warnings, 0 infos

Description	Resource	Path	Location
Errors (1 item)			
Target line number does not exist (70)	Hello.zbi	My Project	line 6

Navigator View

This View displays the list of file and objects associated with Projects. The **Navigator View** is the primary location for managing the items used in a project.

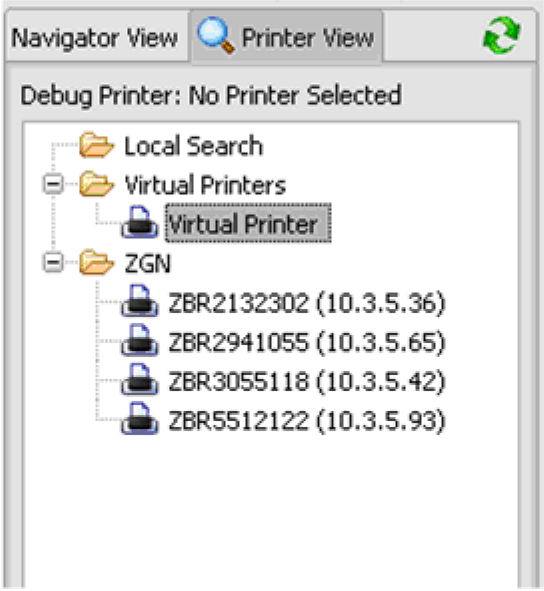


A Project in the **Navigator View** can be open or closed by right-clicking over the Project and selecting the desired action.

Additionally, actions from the **File** menu are also available by right-clicking on the **Navigator View**. The Properties for a Project can also be accessed by right-clicking over a Project name.

Printer View

This View displays the list of printers available for use with Projects. The **Printer View** is the primary location for managing the Printers that will be used with ZBI-Developer.



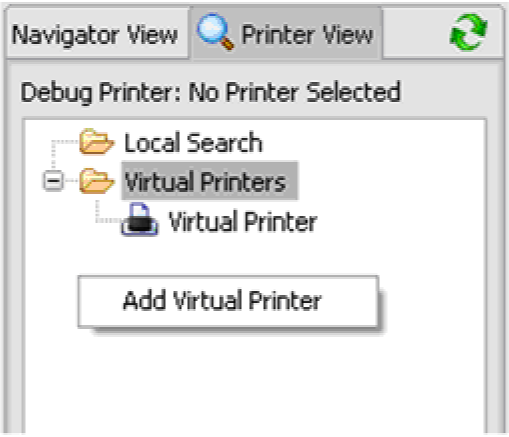
Printers can be connected directly via serial or parallel ports, or a ZebraLink network interface, such as a ZebraNet 10/100 Wired or Wireless print server.

When printers are connected to the PC via a serial, parallel, USB or ZebraLink Ethernet interface, a Debug connection can be made to the printer. This feature allows programs to be run in the Debug mode, and allows the output of the ZBI programs to be sent directly to the printer.

Virtual Printers can also be created so that programs can be run in the Virtual ZBI environment. When a Virtual Printer is used, the output of the program can be directed to a PC Comm Window that displays in the Debug View, or the output can be directed to a serial or parallel port on the PC.

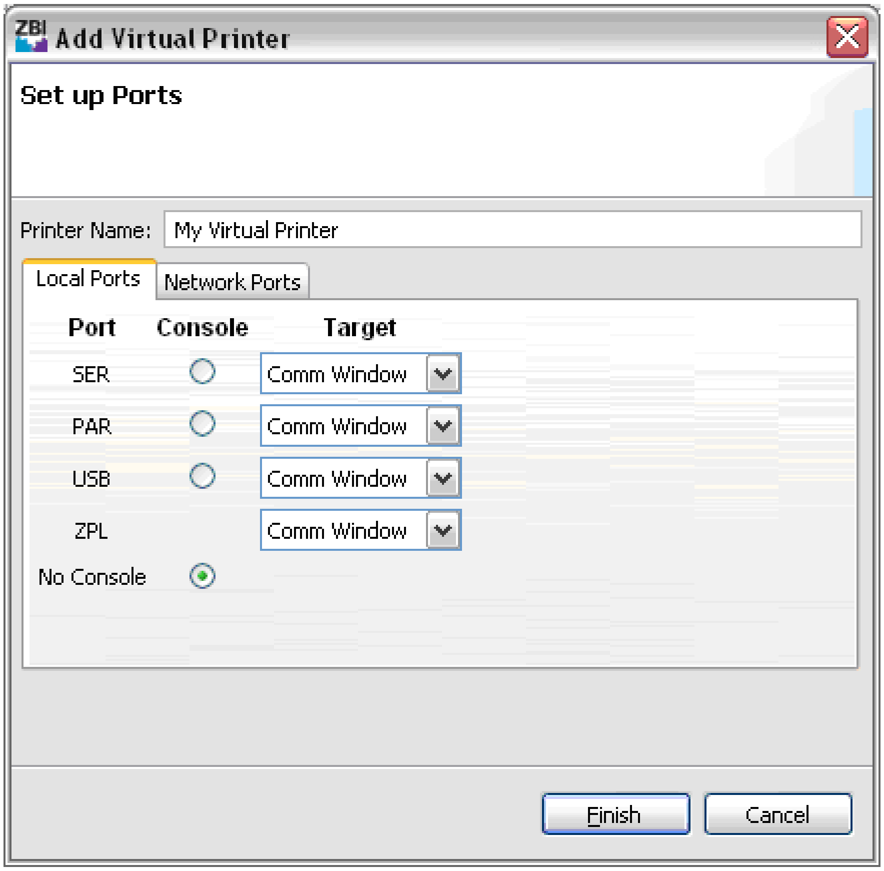
Create a Virtual Printer

ZBI-Developer comes with a Virtual Printer pre-installed. Additional Virtual Printers can be defined by right-clicking over the Virtual Printer folder and selecting **Add Virtual Printer**.



Debug Printer: No Printer Selected

The **Add Virtual Printer** Wizard will be displayed. Enter a Printer name in the textbox. The name can be up to 100 characters long.

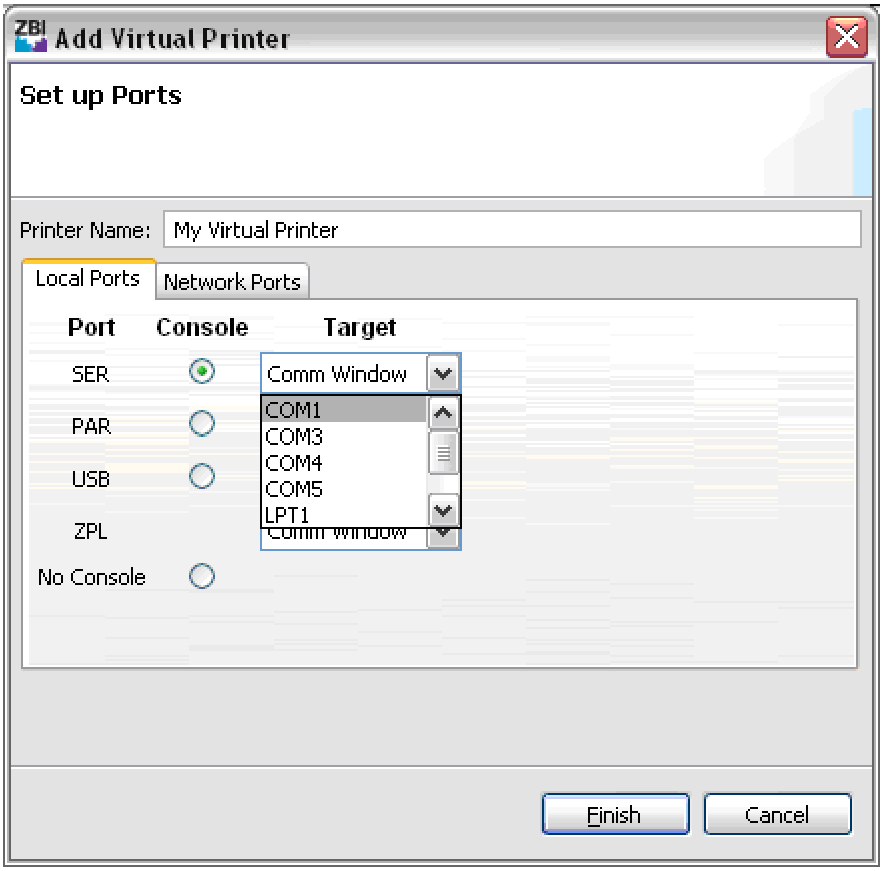


Set up Ports

Printer Name: My Virtual Printer

Port	Console	Target
SER	○	Comm Window
PAR	○	Comm Window
USB	○	Comm Window
ZPL		Comm Window
No Console	●	

Configure where output from the Virtual Printers ports should be directed. Since the printer is Virtual, output can be redirected to one of several locations. For example, if the ZBI program directs that the output should go to a serial port (SER in ZBI commands), the Virtual Printer can redirect that output to a Ports View dialog in the Debug Perspective, or to a physical port on the computer.



Add Virtual Printer

Set up Ports

Printer Name: My Virtual Printer

Port	Console	Target
SER	●	Comm Window
PAR	○	
USB	○	
ZPL		Comm Window
No Console	○	

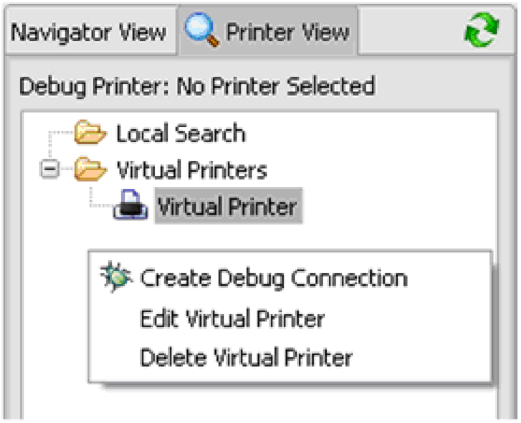
Using this method, a ZBI program could logically send output to a serial port (SER in ZBI command), but that data could be redirected to a parallel port on the PC. In this way, it is possible to test the output of a ZBI program on a printer when a ZBI-Enabled printer is not available.

Additionally, output could be redirected to a Ports View dialog on the PC instead of to a physical communication port. Using this method a ZBI program can be created and tested when no printer is available.

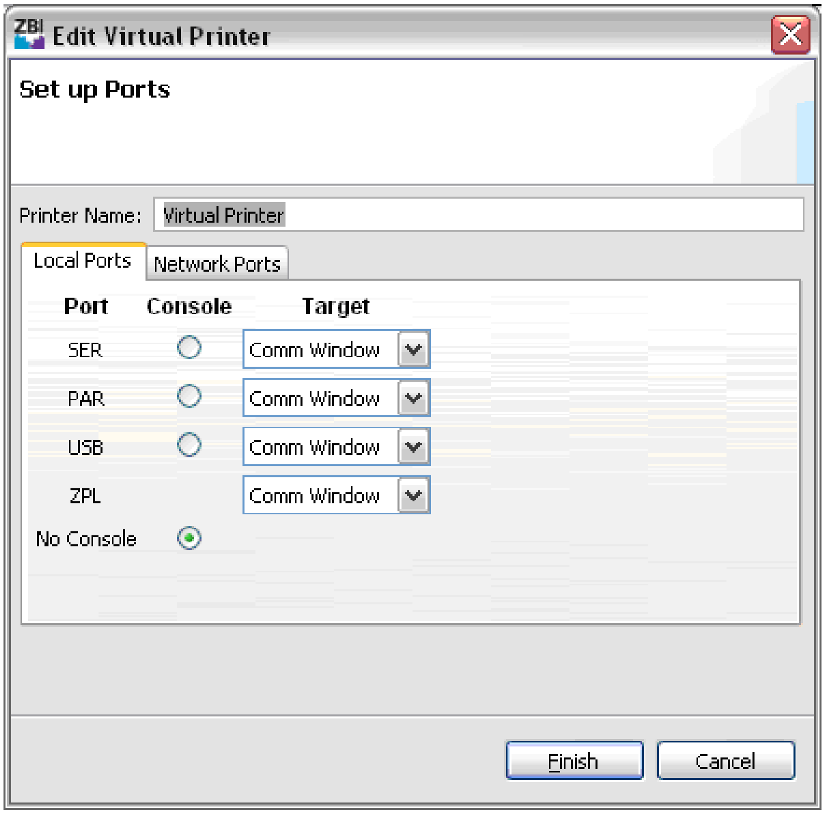
> **Note •** Virtual Printers can be used to examine the output of a ZBI program. They do not have the ability to create an image of a printed label based on the receipt of printer commands. That capability is only present in the printer.

Edit a Virtual Printer

Virtual Printers can be easily edited to alter their settings. To Edit a **Virtual Printer,** right-click over the **Virtual Printer** in the **Printers View** and select **Edit Virtual Printer**.



The **Edit Virtual Printer** Wizard will display.

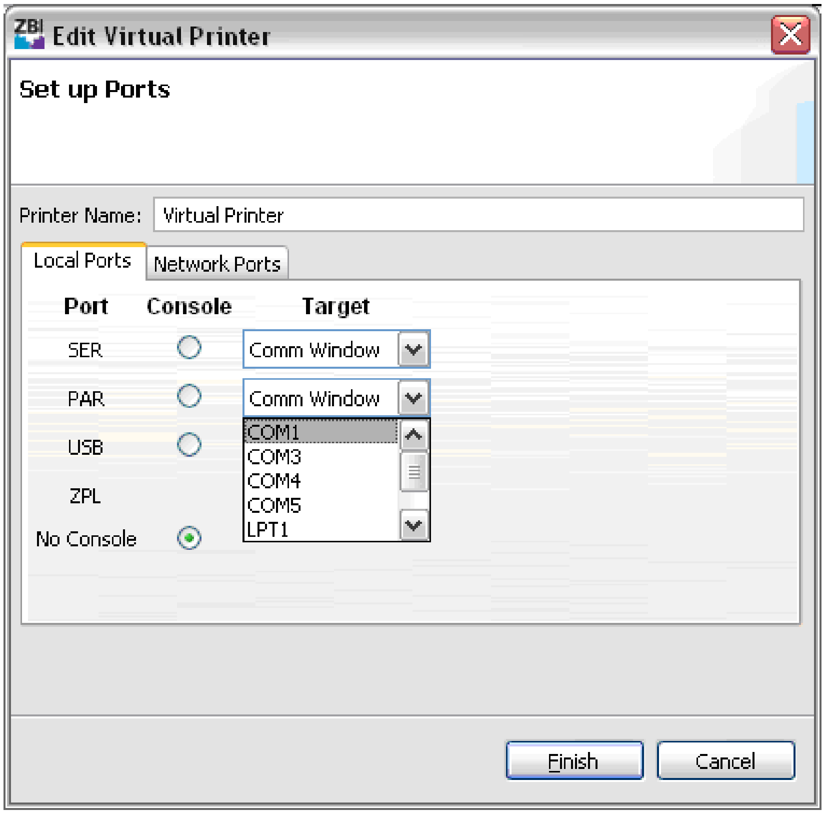


Set up Ports

Printer Name: Virtual Printer

Port	Console	Target
SER	○	Comm Window
PAR	○	Comm Window
USB	○	Comm Window
ZPL		Comm Window
No Console	●	

Configure where output from the Virtual Printers ports should be directed. Since the printer is Virtual, output can be redirected to one of several locations. For example, if the ZBI program directs that the output should go to a parallel port (PAR in ZBI commands), the Virtual Printer can redirect that output to a Ports View dialog in the Debug Perspective, or to a physical port on the computer.



Edit Virtual Printer

Set up Ports

Printer Name: Virtual Printer

Port	Console	Target
SER	○	Comm Window
PAR	○	Comm Window
USB	○	
ZPL		
No Console	●	

Using this method, a ZBI program could logically send output to a parallel port (PAR in ZBI command), but that data could be redirected to a serial port on the PC. In this way, it is possible to test the output of a ZBI program on a printer when a ZBI-Enabled printer is not available.

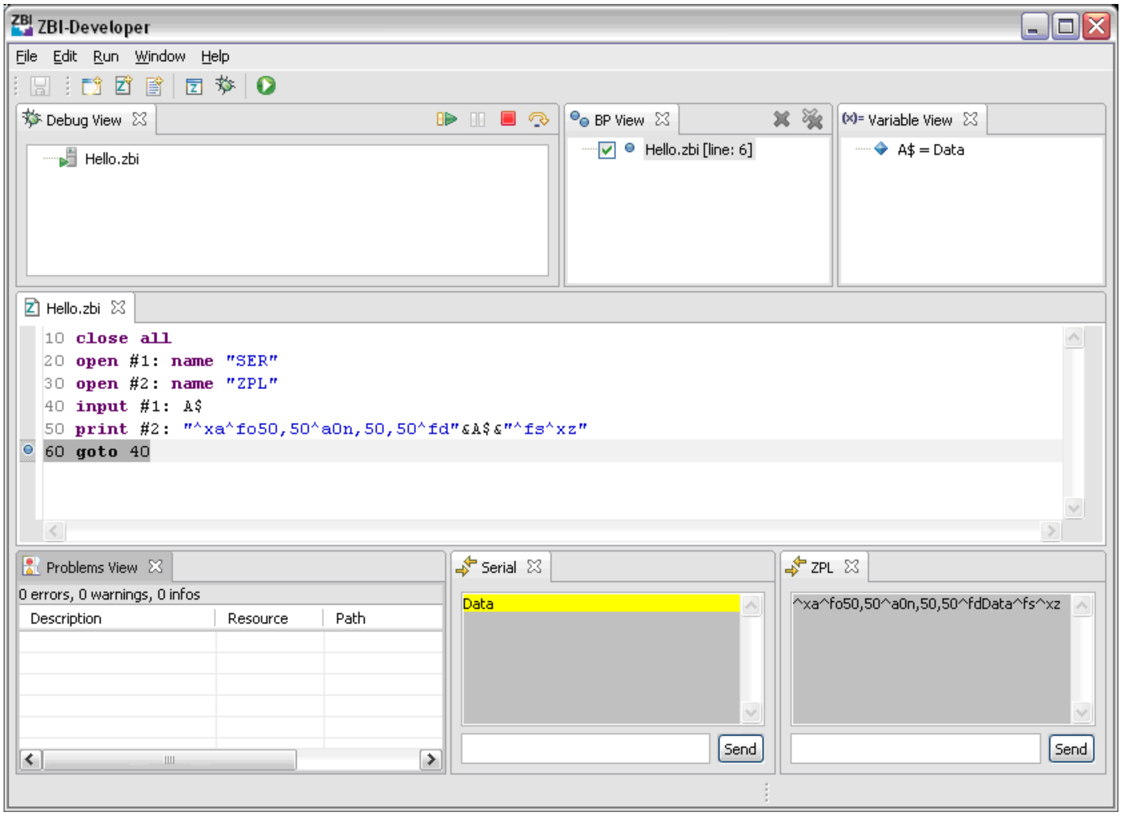
Additionally, output could be redirected to a Ports View dialog on the PC instead of to a physical communication port. Using this method, a ZBI program can be created and tested when no printer is available.

> **Note •** Virtual Printers can be used to examine the output of a ZBI program. They do not have the ability to create an image of a printed label based on the receipt of printer commands. That capability is only present in the printer.

Using Virtual Printers

The figure below illustrates the testing of a simple ZBI program using a Virtual Printer.

In this case, the ZBI program receives data from the Serial port and sends output to the ZPL print engine in the printer. The Virtual printer is configured to display both the Serial port input and ZPL port output in Ports View dialogs. A **Breakpoint** has been set to cause the program to display the value of the A$ input variable in the Variable View dialog.



Debug View: Hello.zbi

BP View: ☑ Hello.zbi [line: 6]

Variable View: A$ = Data

```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

Problems View: 0 errors, 0 warnings, 0 infos

Description	Resource	Path

Serial: Data

ZPL: ^xa^fo50,50^a0n,50,50^fdData^fs^xz

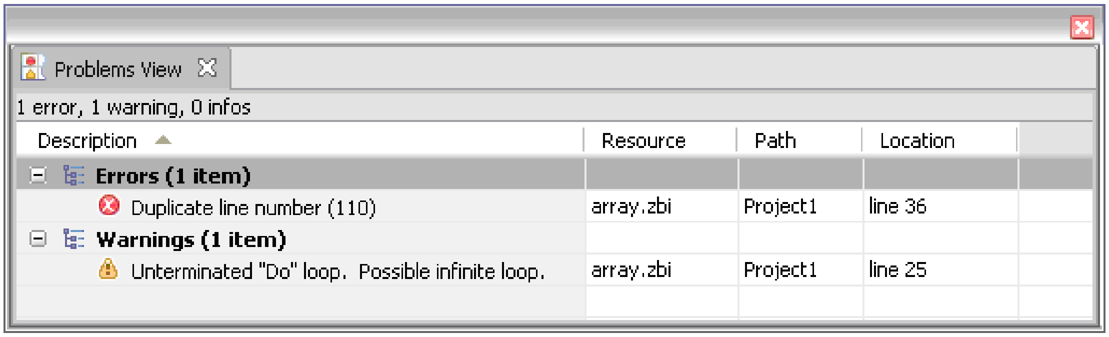
Note • When using the **Debug Perspective**, ZBI-Developer can be occupied while it is executing the program. If you are done using the **Debug Perspective**, be sure to click **Terminate**.

Problems View

This View displays the list of **Warnings** and **Errors** the software has identified in the ZBI programs in all open Projects. If a project is closed, **ZBI-Developer** will not display **Problems** for the project. Only files with a ".zbi" extension will be checked for issues.

The **Problems View** uses plain language to inform the user of **Warnings** and **Errors** that ZBI-Developer has identified in a ZBI programs. Double-clicking on an error will cause the program to automatically highlight the line in the ZBI program that has been identified. If the program is closed, it will be opened in the **ZBI Edito**r.

ZBI programs that contain **Warnings** and **Errors** can still be run on a **Virtual Printer**, tested in the **Debug Perspective,** or sent to a printer. The **Problems View** exists only to inform about issues, not to stop users from writing and using programming techniques.



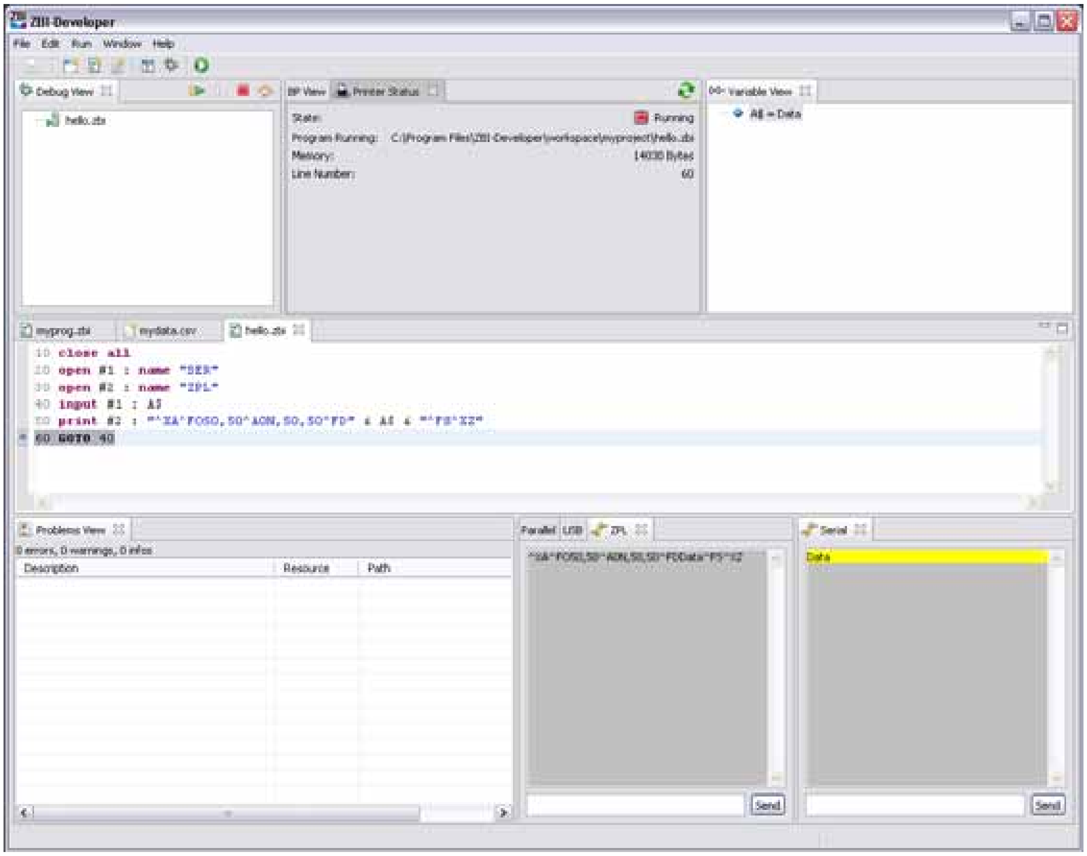
1 error, 1 warning, 0 infos

Description	Resource	Path	Location
⊟ **Errors (1 item)**			
Duplicate line number (110)	array.zbi	Project1	line 36
⊟ **Warnings (1 item)**			
Unterminated "Do" loop. Possible infinite loop.	array.zbi	Project1	line 25

Debug Perspective

The **Debug Perspective** is used to execute a ZBI program, either with a physical printer, or with a Virtual Printer. By using the Debug View, the programmer can quickly test the logic and flow of their ZBI program.

There are several key Views in the **Debug Perspective.**



State: Running
Program Running: C:\Program Files\ZBI-Developer\workspace\myproject\hello.zbi
Memory: 14030 Bytes
Line Number: 60

A$ = Data

```
10 close all
20 open #1 : name "SER"
30 open #2 : name "ZPL"
40 input #1 : A$
50 print #2 : "^XA^FO50,50^A0N,50,50^FD" & A$ & "^FS^XZ"
60 GOTO 40
```

0 errors, 0 warnings, 0 infos

Description	Resource	Path

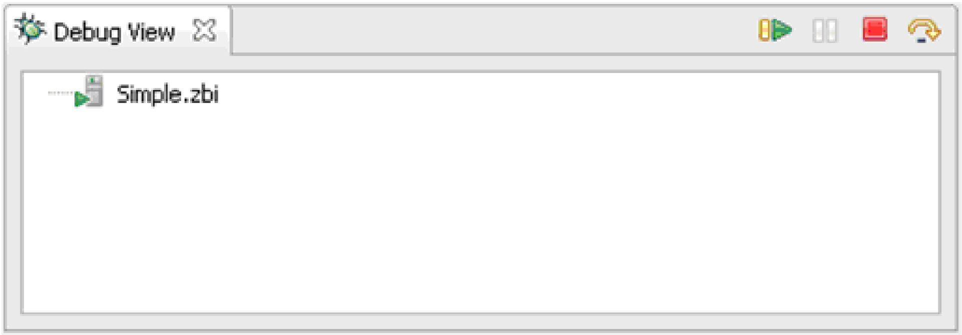
```
^XA^FO50,50^A0N,50,50^FDData^FS^XZ
```

```
Data
```

Debug View

The **Debug View** displays the name of the program being tested.

Button	Description
▶	The **Run/Debug Application** button (located in the **Toolbar**) begins the execution of the program.
⏸	The **Pause** button (located on the **Debug View**) suspends the execution of the program so that the user can browse the program, check the printer, examine the **Variable View**, etc.
🟥	The **Terminate** button halts the execution of the program.
↷	The **Step Over** button steps over a set breakpoint. Program execution will continue on the next line.
⏯	The **Resume** button resumes a paused program execution.

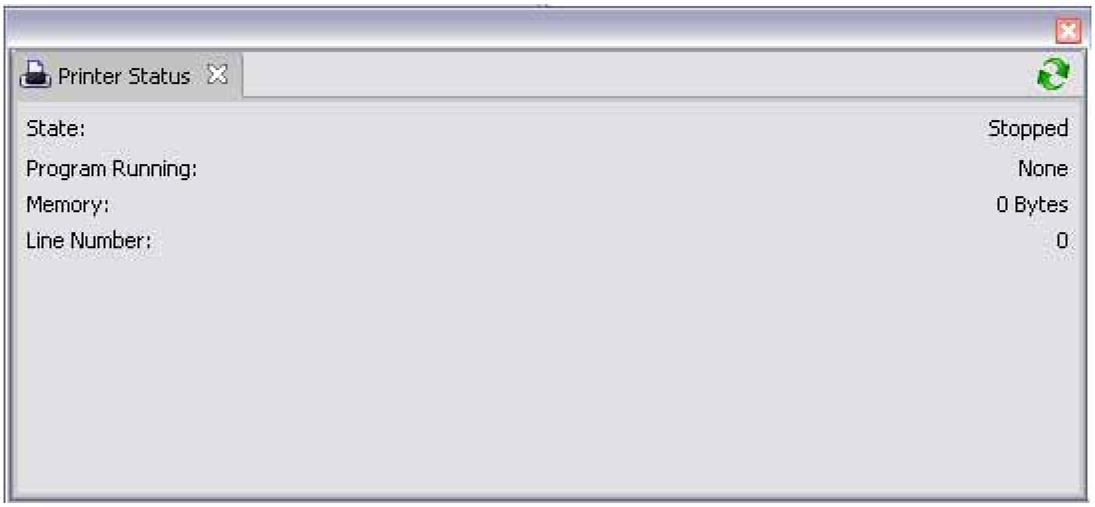


Note • When using the **Debug Perspective,** ZBI-Developer can be occupied while it is executing the program. If you are done using the **Debug Perspective**, be sure to click **Terminate.**

Printer Status View

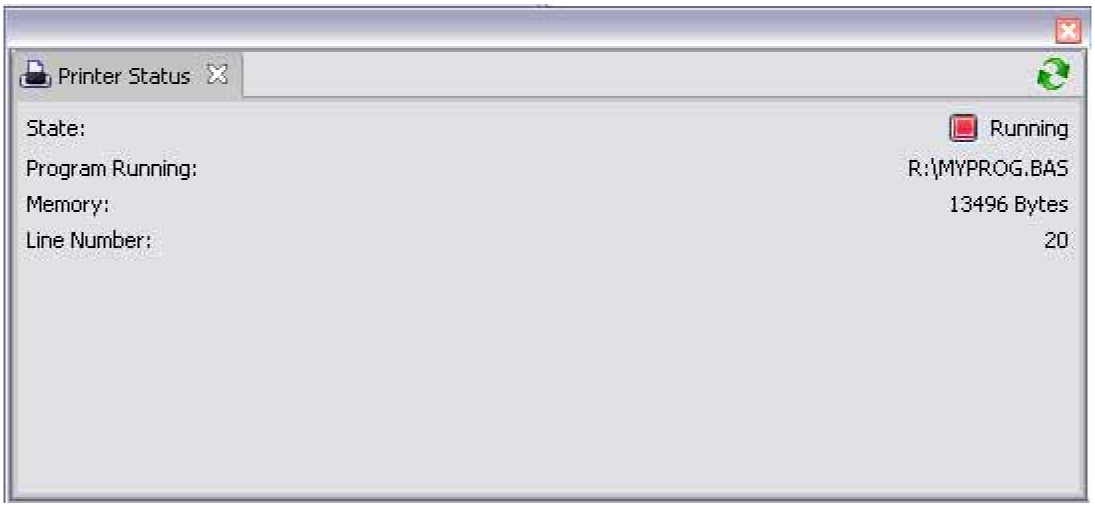
The **Printer Status View** displays the status of the current debug printer.

Status Detail	Description
State	Indicates if ZBI is 'Running' or 'Stopped' on the debug printer.
Program Running	Indicates the name of which program is running on the debug printer.
Memory	Indicates how much memory, in bytes, is being consumed by running the ZBI program.
Line Number	Indicates which line of the currently running program is being executed.



State: Stopped
Program Running: None
Memory: 0 Bytes
Line Number: 0

When a program is **Stopped**, the **Refresh** button ⟳ can be used to update the status. This can be pressed regardless if a program was started in the ZBI IDE.



State: 🟥 Running
Program Running: R:\MYPROG.BAS
Memory: 13496 Bytes
Line Number: 20

When a program is **Running** the terminate button 🟥 can be pressed to stop the program. This can be done for any ZBI program running on the debug printer, even those not started with the ZBI IDE.

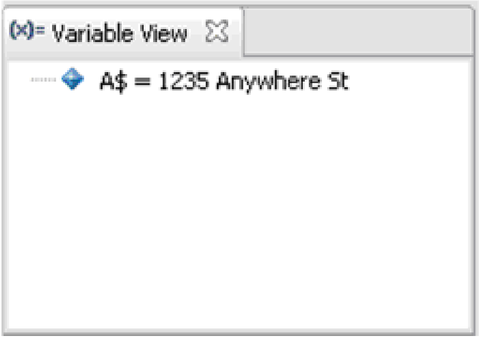
Pressing the **Refresh** button 🔄 will refresh the status of the debug printer.

(x)= Variable View

Displays the variables currently present in a ZBI program during execution in the **Debug View.**

The **Variable View** makes it simple to identify the contents of the variables in your program. When combined with the **Port Views** and carefully set breakpoints, the programmer can type information in and immediately see how data is being populated into variables in a program.

> **Note •** In some cases, Variables may not appear in the Variable View. This can happen when no Breakpoints have been set in the program. Be sure to set Breakpoints to make the best use of the **Variable View** feature.



A$ = 1235 Anywhere St

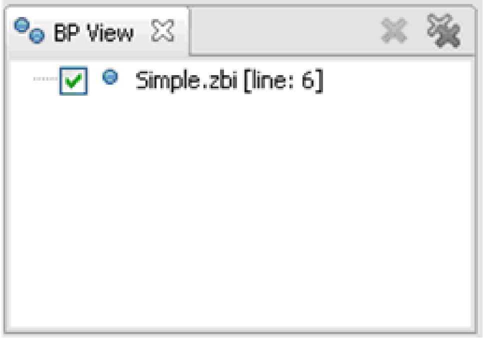
Breakpoints View

Displays the Breakpoint set in a currently running ZBI program. To set a Breakpoint in the ZBI program, right-click immediately to the left of a line of ZBI code and choose **Toggle Breakpoint**.

Enabled breakpoints are noted with a blue circle ● . Disabled breakpoints are noted with a white circle ○ . Breakpoints are displayed in the vertical editor ruler and in the **Breakpoints View.**

A breakpoint suspends the execution of a program at the location where the breakpoint is set. In the **ZBI Editor**, breakpoints can be enabled and disabled by using the displayed checkboxes. Double-clicking on a breakpoint in the **Breakpoints View** will highlight it in the **ZBI Editor**.

Note • The maximum number of breakpoints per program is 16.



☑ ● Simple.zbi [line: 6]

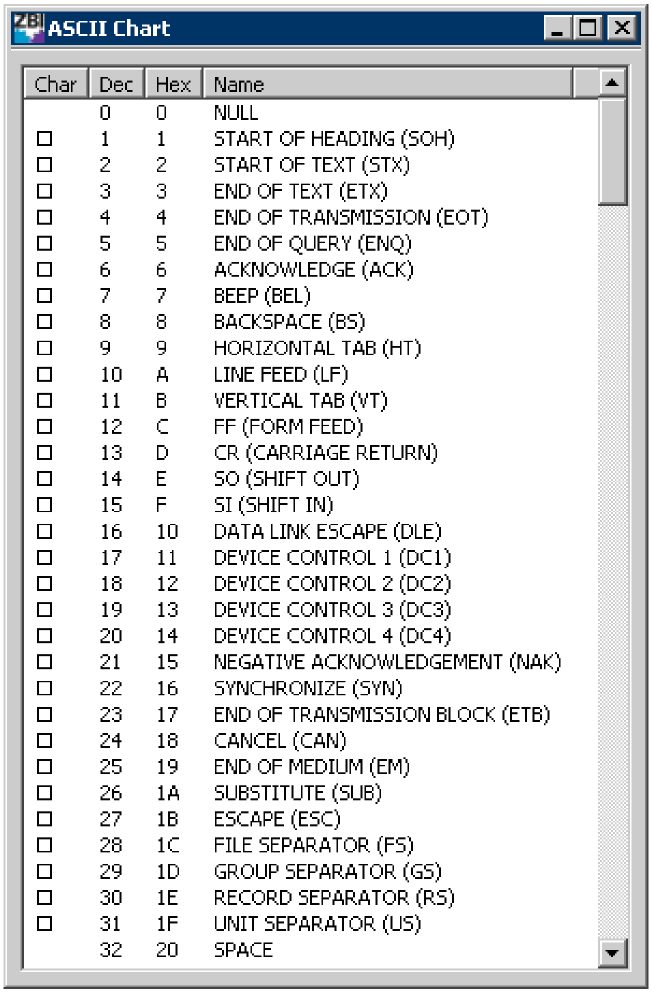
Help Menu

Update

This menu selection causes ZBI-Developer to contact the web-based program update service and obtain the most recent update. The program will not be updated if it is the most current version. Follow the prompts to complete the update.

View ASCII Table

Causes an ASCII chart to be displayed as shown below. Scroll on the chart to view desired character(s).

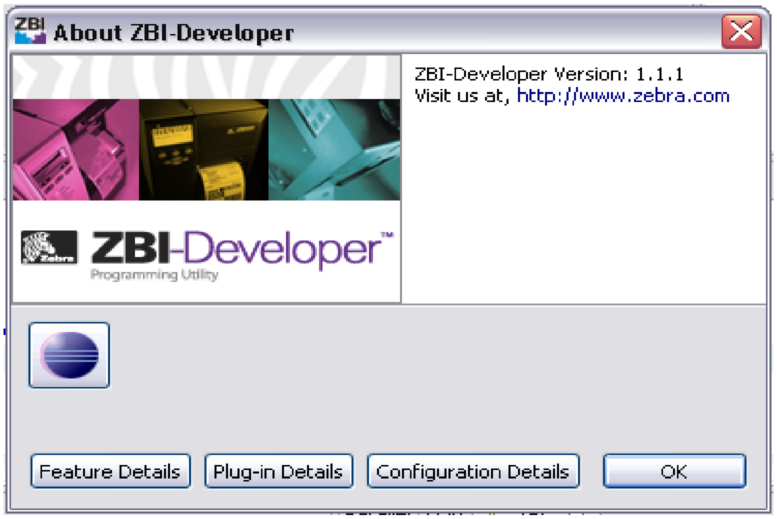


ASCII Chart

Char	Dec	Hex	Name
	0	0	NULL
□	1	1	START OF HEADING (SOH)
□	2	2	START OF TEXT (STX)
□	3	3	END OF TEXT (ETX)
□	4	4	END OF TRANSMISSION (EOT)
□	5	5	END OF QUERY (ENQ)
□	6	6	ACKNOWLEDGE (ACK)
□	7	7	BEEP (BEL)
□	8	8	BACKSPACE (BS)
□	9	9	HORIZONTAL TAB (HT)
□	10	A	LINE FEED (LF)
□	11	B	VERTICAL TAB (VT)
□	12	C	FF (FORM FEED)
□	13	D	CR (CARRIAGE RETURN)
□	14	E	SO (SHIFT OUT)
□	15	F	SI (SHIFT IN)
□	16	10	DATA LINK ESCAPE (DLE)
□	17	11	DEVICE CONTROL 1 (DC1)
□	18	12	DEVICE CONTROL 2 (DC2)
□	19	13	DEVICE CONTROL 3 (DC3)
□	20	14	DEVICE CONTROL 4 (DC4)
□	21	15	NEGATIVE ACKNOWLEDGEMENT (NAK)
□	22	16	SYNCHRONIZE (SYN)
□	23	17	END OF TRANSMISSION BLOCK (ETB)
□	24	18	CANCEL (CAN)
□	25	19	END OF MEDIUM (EM)
□	26	1A	SUBSTITUTE (SUB)
□	27	1B	ESCAPE (ESC)
□	28	1C	FILE SEPARATOR (FS)
□	29	1D	GROUP SEPARATOR (GS)
□	30	1E	RECORD SEPARATOR (RS)
□	31	1F	UNIT SEPARATOR (US)
	32	20	SPACE

About ZBI-Developer

Causes the **About ZBI-Developer** dialog to display.



ZBI-Developer Version: 1.1.1
Visit us at, http://www.zebra.com

Configuration of ZBI-Developer

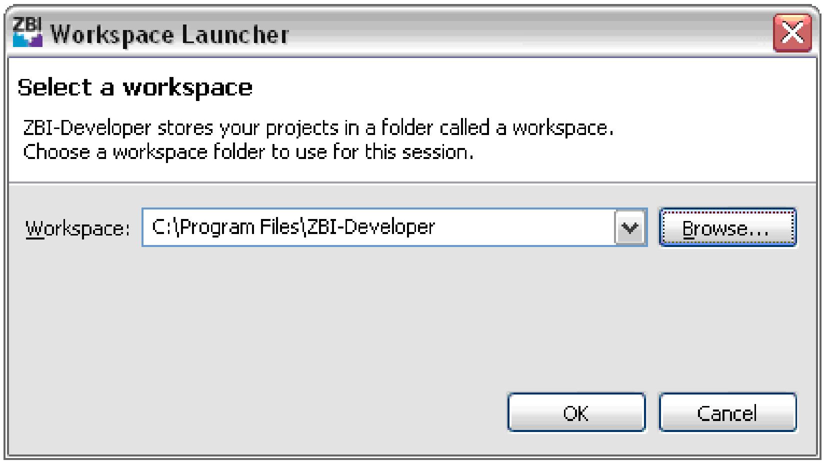
This section describes you how to set up and customize **ZBI-Developer** to your preferences.

Changing the Workspace Location

By default, **ZBI-Developer** stores its files in a Workspace folder, or directory on the PC it is being used on. Knowing where this directory is makes it possible to ensure that the directory is being backed up on a regular basis (a recommended practice!).

Additionally, for some work-groups, it may be desirable to maintain the Workspaces of several ZBI Developers on networked server rather than on local drives. This practice is also encouraged, especially if it means that the Workspace directories will be regularly archived

To change the Workspace location, go to File menu and select **File > Switch Workspace**. The **Workspace Launch** wizard will display, showing the current location of the workspace. Click on the **Browse** button to navigate to and define the new location to store the Workspace in.



Workspace Launcher

Select a workspace

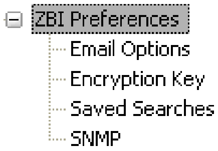
ZBI-Developer stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: C:\Program Files\ZBI-Developer

Note • Once a new Workspace location has been defined, ZBI-Developer will take a few moments to move the necessary files and restart itself. You will see the start-up splash screen as the program restarts.

Setting Preferences

There are four types of Preferences to configure for ZBI-Developer. To begin setting Preferences, select **Window** > **Preferences**. When the Preferences dialog appears, click on the "+" symbol next to **ZBI Preferences** to expand the list.



Email Options

The Email Options control the settings used to send email from ZBI Programs when a Virtual Printer is being used. Enter a Server Name specific to the site the program will be used in and set the Port to use as needed. Port 25 is the default port used for Email.

Encryption Key

This dialog controls the location and name of the Encryption Key ZBI-Developer will use when the user chooses to use the optional ZBI program Encryption feature.

Note • It is recommended that one key be used for entire company or development group. While it is possible to create multiple keys (i.e., one for each project, or one for every developer in a given company, this is not recommended to decrease the possibility of lost keys.

Saved Searches

In some cases, the desired printer may not be automatically discovered. This can happen because the printer is on a different subnet, or because traffic on the network is being controlled in such a way that the discovery process is not allowed to complete. Many times, what is needed is a different type of search.

ZBI-Developer offers five types of network searches:

- Local Broadcast Search
- Subnet Search
- Directed Broadcast Search
- Multicast Search
- IP Address Search

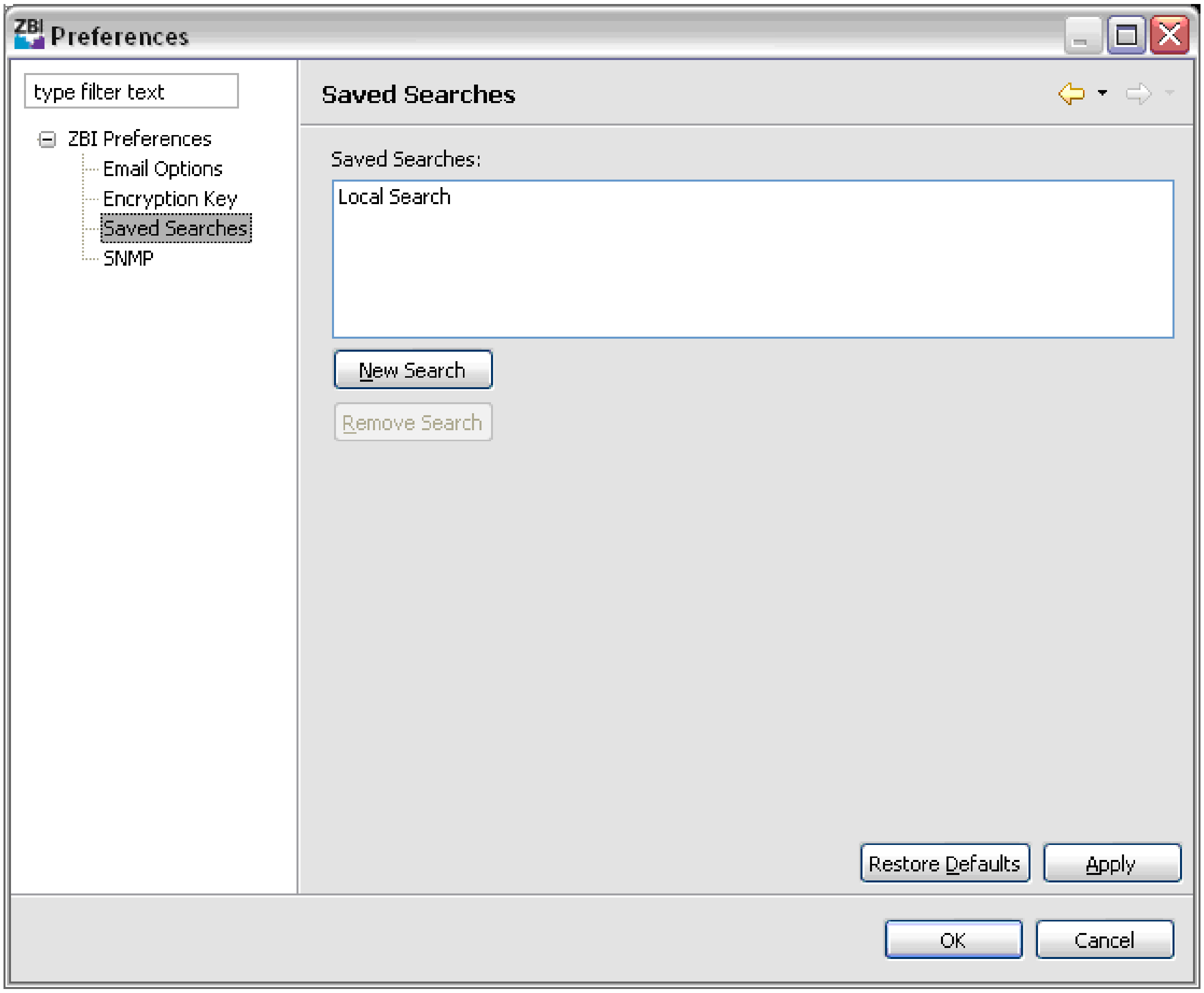
Note • The connection to a networked printer can be tested by right-clicking over the printer in the **Printers View** and choosing Print Configuration label. If the connection is active, a configuration label should print.

To set up a search for a network printer, do the following:

1. Go to the menu and choose **Window** > **Preferences**. The Preferences dialog will be displayed.

2. Click on the "+" symbol next to **ZBI-Preferences.**

3. Click on **Saved Searches**.



4. Click on **New Search**.

Local Broadcast Search

Local broadcasts send a discovery request to the local broadcast address of 255.255.255.255. This request finds printers in the same local network as the requesting computer. The local broadcast search is the default search type used by the program.

To create a local broadcast search, do the following:

1. In the Search Type dialog, select **Local Broadcast**. The Value fields are automatically populated.

2. Type a name in the Search Group Name textbox.

3. Click **Add**. The Value field is added to Search type.
4. Click **OK** to accept the new Search Type.
5. Click **OK** to exit the Preferences dialog.

Subnet Search

The Subnet Search sends a discovery request to each address in the specified subnet. This feature is useful for networks that prevent broadcast packets from passing across routers. The Subnet search requires you to type in the first three subnet octets.

Example • A request to 192.168.2.* sends a request to: 192.168.2.1 through 192.168.2.254.

To create a Subnet Search, do the following:

1. From the Search Type dialog, select **Subnet**.
2. Type a name in the Search Group Name textbox.
3. In the Value field, type the first three subnet octets.
4. Click **Add**.
5. Click **OK** to accept the new Search Type.
6. Click **OK** to exit the Preferences dialog.

Directed Broadcast Search

Directed broadcasts send a discovery request to the broadcast address of the specified subnet.

Note • If a broadcast search does not work, the broadcast packets might be disabled across routers. Try Multicast or Subnet search.

Example • The directed broadcast search requires you to type in the first three subnet octets, followed by 255, such as 192.168.2.255.

To create a Directed Broadcast search, do the following:

1. In the Search Type dialog, select **Directed Broadcast**. The fourth octet of the Value field is automatically populated.
2. In the Value field, type the first three subnet octets.
3. Type a name in the **Search Group Name** textbox.
4. Click **Add**.
5. Click **OK** to accept the new Search Type.
6. Click **OK** to exit the Preferences dialog.

Multicast Search

Multicast uses the multicast address to send a discovery address to all Zebra printers across as many routers as the hop count specifies. A multicast search may return a large number of printers in some installations.

This feature might be disabled on some routers and switches. See your network administrator to determine if a search failure was due to this feature being disabled.

To create a multicast search, do the following:

1. In the Search Type dialog, select **Multicast (Hops)**.
2. In the **Value** field, type the number of hops desired. The recommended starting point is 5.
3. Type a name in the **Search Group Name** textbox.
4. Click **Add**.
5. Click **OK** to accept the new Search Type.
6. Click **OK** to exit the Preferences dialog.

IP Address Search

To create a search for an IP address, do the following:

1. In the Search Type dialog, select **IP Address xxx.xxx.xxx.xxx**.

   Example • You would type in an IP address, such as 10.3.4.97.

2. In the **Value** field, type in an IP address.
3. Type a name in the **Search Group Name** textbox.
4. Click **Add**.
5. Click **OK** to accept the new Search Type.
6. Click **OK** to exit the Preferences dialog.

SNMP

The SNMP settings are used to control how ZBI-Developer interacts with a printer when the on-printer debugging features are in use. These features are available when a printer running firmware v60.16.x and v53.16.x or later are in use. ZBI-Developer will use SNMP commands ("set" and "get") to activate the debugging features on the printer when the indicated firmware versions are being used. The default setting are the recommended settings.

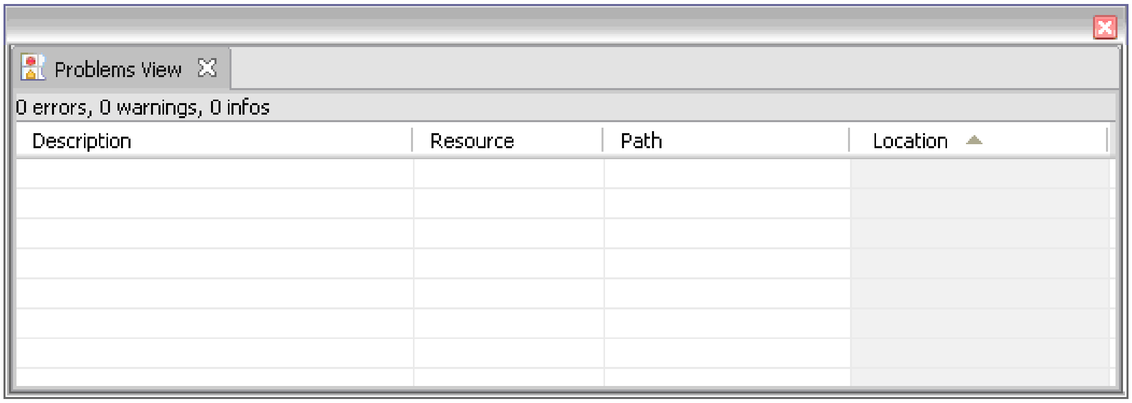
> **Note •** Some networks may be configured to not allow SNMP traffic. Please consult with your network administrators first if the on-printer debugging features are not working when you have successfully connected to a printer. The connection to a networked printer can be tested by right-clicking over the printer in the **Printers View** and choosing **Print Configuration label**. If the connection is active, a configuration label should print.

Changing the Screen Layout

The initial screen layout may be appropriate for many uses, however the user may find that the layout is not optimized for the project being worked on. The layout of many of the **Views** can be changed to suit the needs of the user.

To move a View, click and drag it to a new location. It is also possible to right-click over the tab for a View and select **Move** > **View**. While a View is being moved, an outline of the View will be visible.

> **Note •** For example, the **Problems View** can be moved or even detached from the main window of the program to maximize the space provided for the **ZBI Editor View**. Here it is shown detached from the main window of the program.



0 errors, 0 warnings, 0 infos

Description	Resource	Path	Location

To re-attach a detached **View**, right-click over it and un-select **Detached**. It will re-join the main window of the program.

ZBI-Developer Tutorial

This Tutorial demonstrates creating, testing and distributing a simple ZBI project, using many of the key features of **ZBI-Developer**.

This is the program that will be created:

```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

Tutorial Line Number	Line-by-Line Explanation
10	Closes all ZBI Ports
20	Opens a `"#1"` port and names it `"SER"`
30	Opens a `"#2"` port and names it `"ZPL"`
40	Receives from `"#1"` port and stores it in the `"A$"` variable
50	Embed the data stored in `"A$"` in ZPL commands and prints it
60	Returns to line 40 to look for more input

In this Tutorial, it will not be necessary to have a ZBI-Enabled printer present. All of the activities described here can be performed without a printer present. However, if a ZPL printer is available, it will augment the Tutorial if you connect the printer to the serial port of the computer running **ZBI-Developer**.

Some key concepts that this Tutorial covers:

- Views
- Perspectives

Views

The software offers several **Views**. These are used to display different information screens.

Perspectives

Two modes of operation, or Perspectives, are available in ZBI-Developer

- **ZBI Perspective** – for writing ZBI programs
- **Debug Perspective** – for testing and checking programs

Project Files

Projects are used to organize and track files associated with a single or set of ZBI programs. Projects can contain several file types:

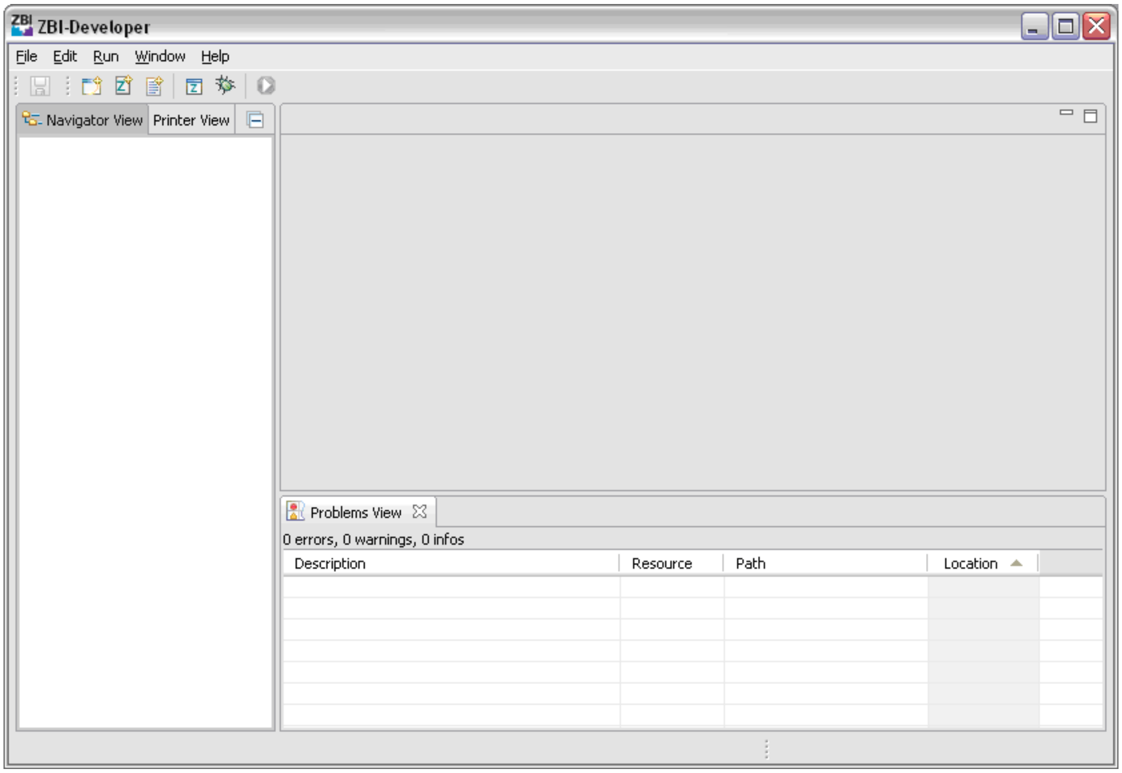
- **ZBI programs** – containing up to an 8 character file name and the .zbi extension. ZBI programs are displayed with the Z icon.
- **Data Files** – these can be any file, ASCII or Unicode encoded, that is associated with the Project. Data Files are displayed with the 📄 icon.
- **Project Key** – the Encryption Key that will be used with the Project. The Project Key file is displayed with the 🔑 icon. Each Project can have its own Project Key file.
- **Encrypted ZBI program** – ZBI programs that have been encrypted are displayed with the Z🔒 icon. An encrypted program can be viewed in plain text in ZBI-Developer but will be sent to the printer in encrypted form.
- **Project Folders –** displayed with the 📂 icon. Project Folders can be copy and pasted.
- **Other File Types –** Other file types can be viewed in the ZBI-Developer environment, as allowed by file associations on the PC being used. Files can also be opened using the ZBI-Developer Text Editing environment. For example, a .jpg file would open using the default system viewer, unless the ZBI-Developer Text Editor is chosen.

Note • It is recommended that Projects and Project files be backed up on a regular basis. ZBI-Developer does not automatically back-up files or offer a method for un-encrypting programs if the Encryption Key is not present.

Note • Text files (with extensions of .TXT) will be stored on the printer rather than executing the ZPL commands stored within the file. Be sure to name your files with a .ZPL extension, if you are trying to execute the ZPL commands. Files with a .ZPL extension will not be stored on the printer.

Starting the Software

Navigate to the shortcut icon – **Start > Programs > ZBI-Developer**. The program's splash screen will be briefly shown as the program starts and then the main interface window will be displayed.



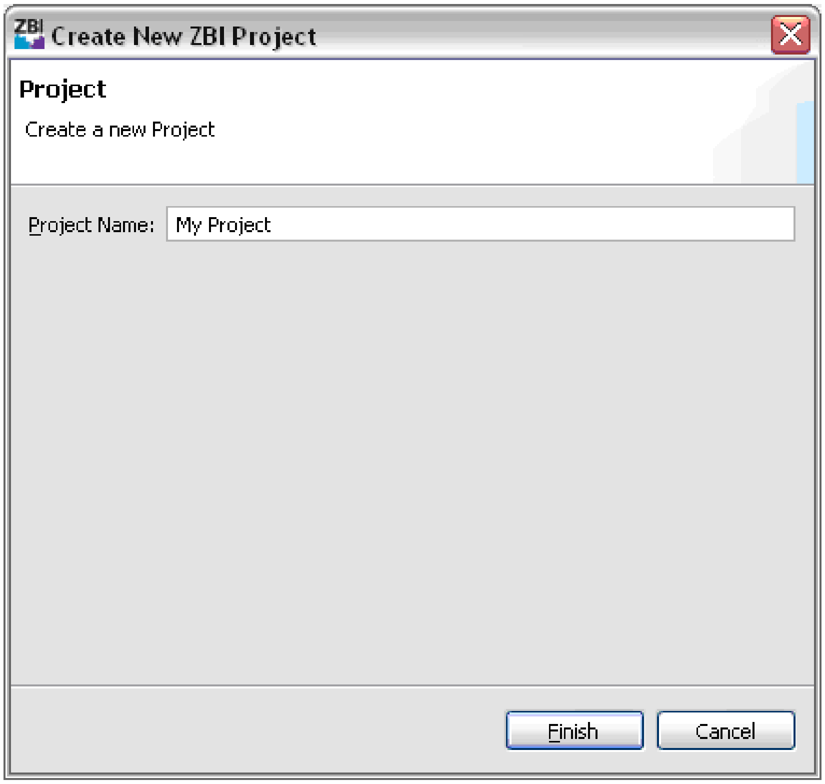
0 errors, 0 warnings, 0 infos

Starting a New Project

To start a new Project, go to **File** > **New** > **New Project.** Alternately, right-click over the **Navigator View** and choose **New** > **New Project.**

The **New Project Wizard** will be displayed.

Name the Project "My Project" and click **Finish.**



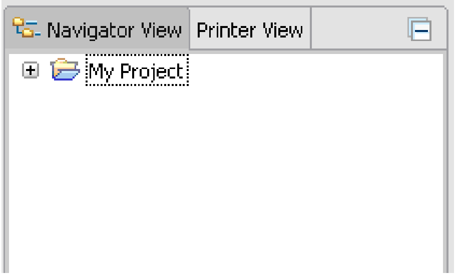
Create New ZBI Project

Project

Create a new Project

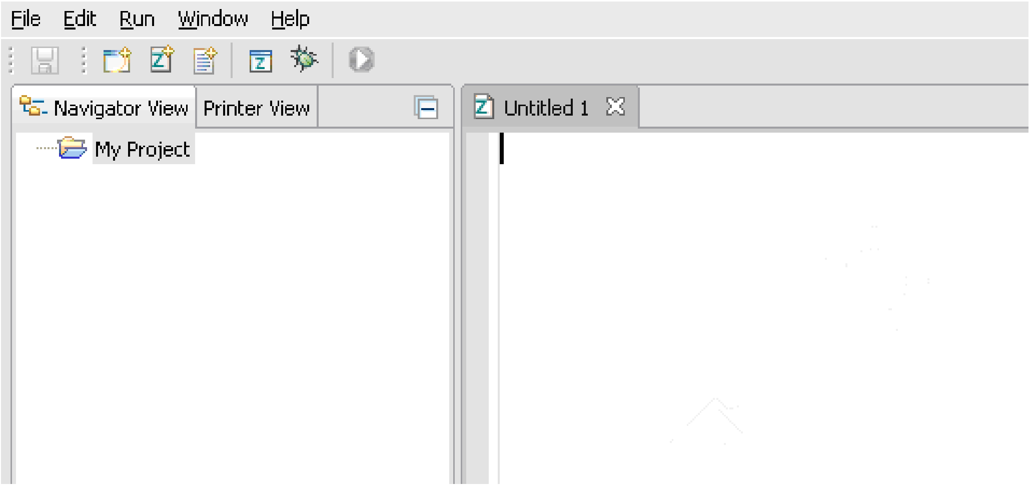
Project Name: My Project

The new Project will be displayed in the **Navigator View**.



Starting a New ZBI Program

Next, start a new ZBI program. Right-click over the "My Project" folder and choose **New** > **New ZBI Program**. A new, blank **ZBI Editor View** will be displayed, with the title "Untitled 1".



Saving a New ZBI Program

To save a new blank ZBI Program:

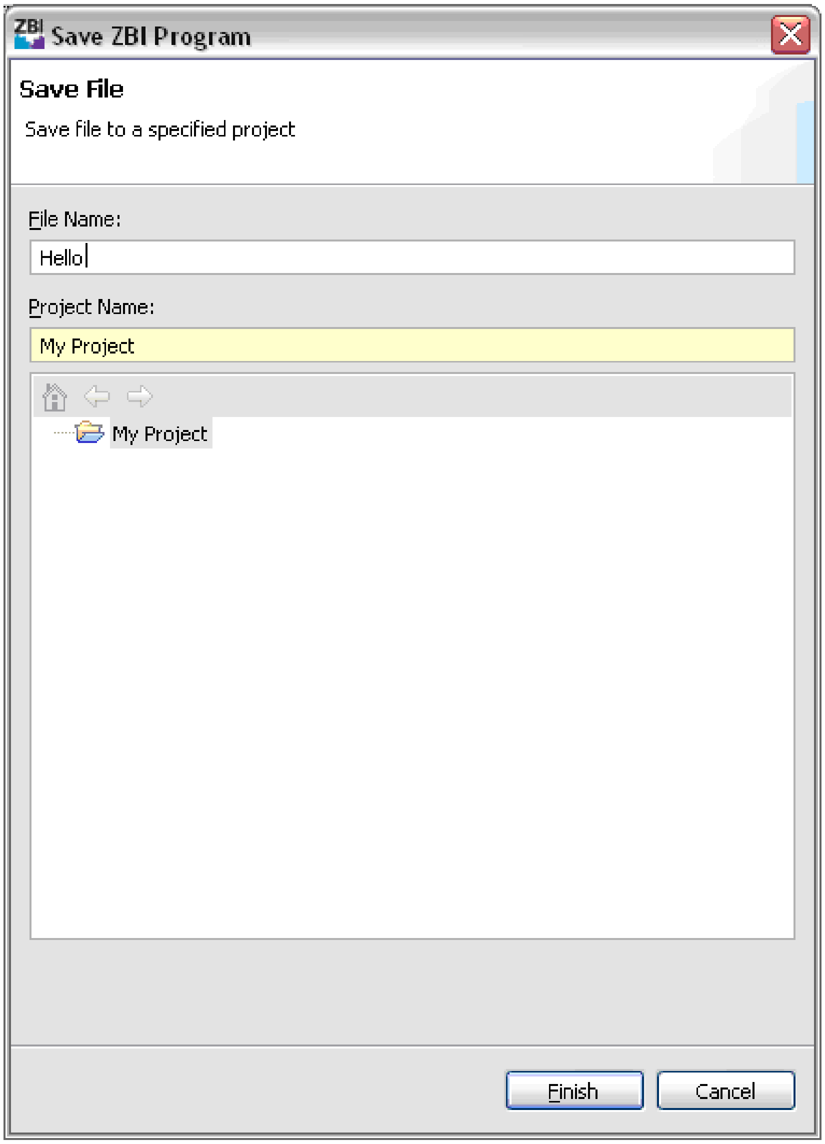
1. Go to **File** > **Save As.** The **Save ZBI Program Wizard** will be displayed.

2. Enter a name for the program - "Hello".

3. Click the **My Project** folder to confirm that it is the Project to save the new program in.

4. Click **Finish**.

Note • It is necessary to click on the Project folder the ZBI program should be saved in. This practice helps keep related files together.



Save ZBI Program

Save File

Save file to a specified project

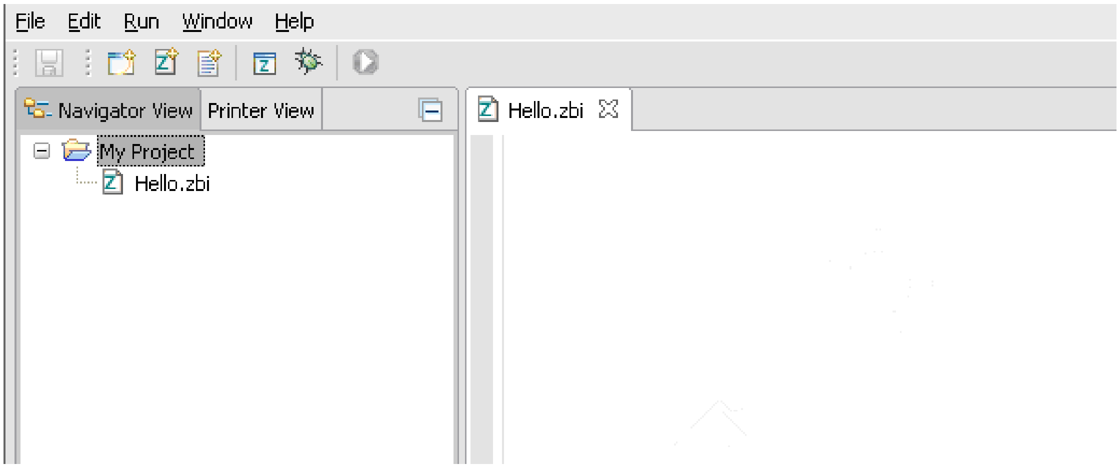
File Name:

Hello

Project Name:

My Project

Once the ZBI program has been saved, the new name will appear in the **ZBI Editor View**. Clicking on the "+" next to the **My Project** folder will cause it to be displayed in the **Navigator View.**



Writing a New ZBI Program

As the ZBI program is begin written, several features of the software can be used to assist the developer.

These include:

- ZBI Command Syntax Help
- Problems View
- Context Sensitive Help
- Virtual Printers
- Debug View

ZBI Command Syntax Help

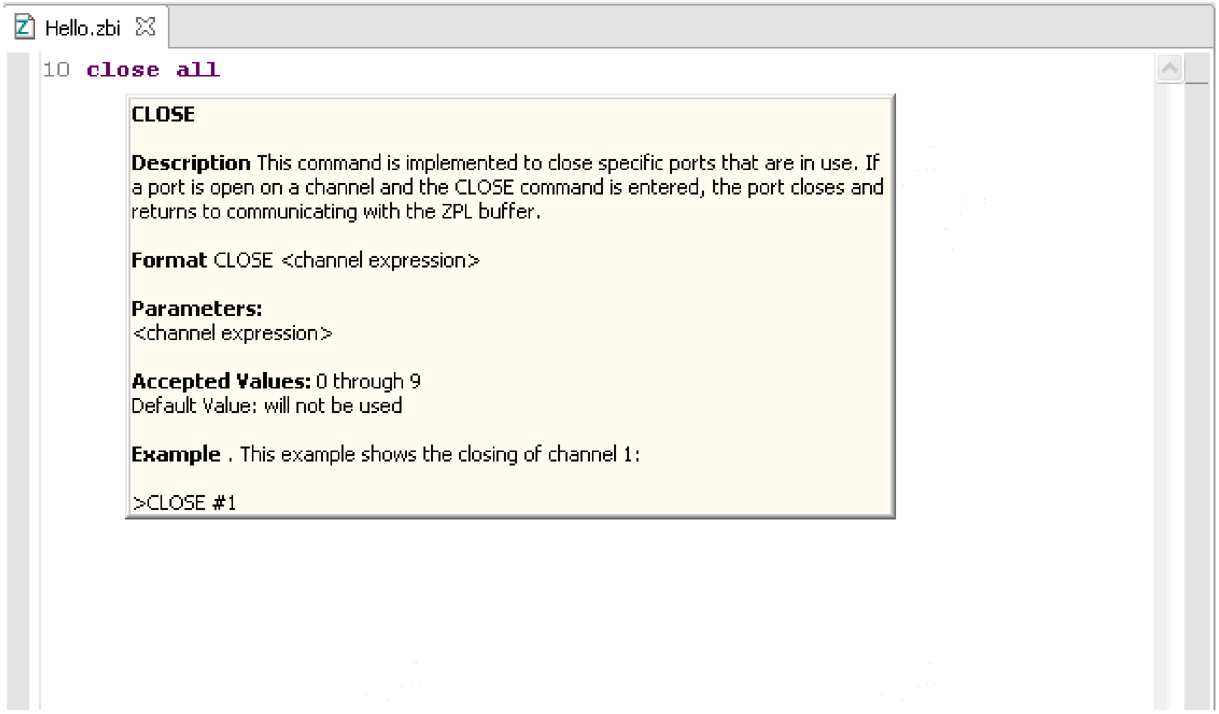
ZBI-Developer can automatically display ZBI syntax command Help by placing the mouse cursor over a command in the **Editor View**.

Write the first line of the program (or copy/paste it from here).

```
10 close all
```

Place the mouse cursor over the word close. After a moment, a dialog will display over the **Editor** showing the correct syntax and usage for the command. This feature can be used with all of the ZBI commands.

Note • The ZBI Command Syntax Help will only appear when a file with a ".zbi" file is being edited.



```
10 close all
```

CLOSE

Description This command is implemented to close specific ports that are in use. If a port is open on a channel and the CLOSE command is entered, the port closes and returns to communicating with the ZPL buffer.

Format CLOSE <channel expression>

Parameters:
<channel expression>

Accepted Values: 0 through 9
Default Value: will not be used

Example . This example shows the closing of channel 1:

>CLOSE #1

The Problems View

The **Problems View** displays the list of **Warnings** and **Errors** the software has identified in the ZBI programs in all open **Projects**. If a project is closed, **ZBI-Developer** will not display **Problems** for the project. Files that do not have a .zbi extension will not be checked for issues.

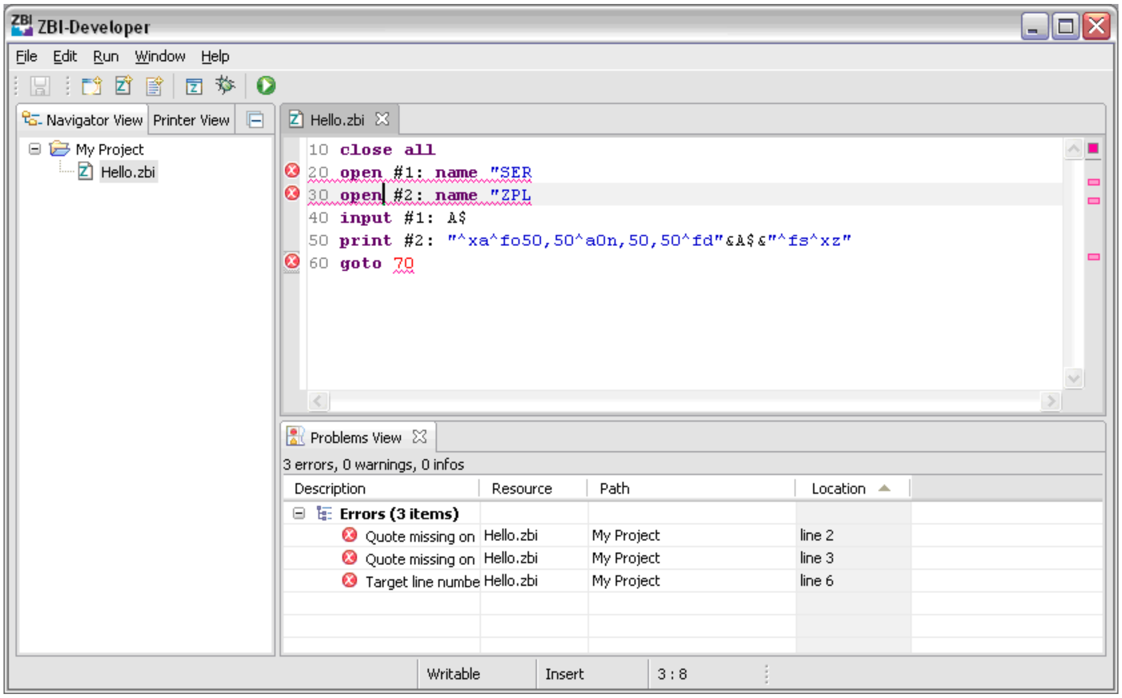
The **Problems View** uses plain language to inform the user of **Warnings** and **Errors** that ZBI-Developer has identified in a ZBI programs. Double-clicking on an error will cause the program to automatically highlight the line in the ZBI program that has been identified. If the program is closed, it will be opened in the **ZBI Edito**r.

Complete the new ZBI program by entering the remainder of the program as shown below.

> **Note •** This content contains intentional errors to help demonstrate a feature of ZBI-Developer.

```
20 open #1: name "SER
30 open #2: name "ZPL
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 70
```

When the program has been entered as shown above, click **File** > **Save** to save the file. Once the program has been saved, **ZBI-Developer** will evaluate the program and display **Errors** or **Warnings** for any lines that contain syntax issues.



```
10 close all
20 open #1: name "SER
30 open #2: name "ZPL
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 70
```

3 errors, 0 warnings, 0 infos

Description	Resource	Path	Location
Errors (3 items)			
Quote missing on	Hello.zbi	My Project	line 2
Quote missing on	Hello.zbi	My Project	line 3
Target line numbe	Hello.zbi	My Project	line 6

Double-clicking on an **Error** or **Warning** in the **Problems View** will cause the portion of the program containing the issue to be highlighted in the **ZBI Editor View**.

Once any Errors or Warnings have been corrected, save the program again to remove the items listed in the **Problems View**. For convenience, here is the corrected version of the program:

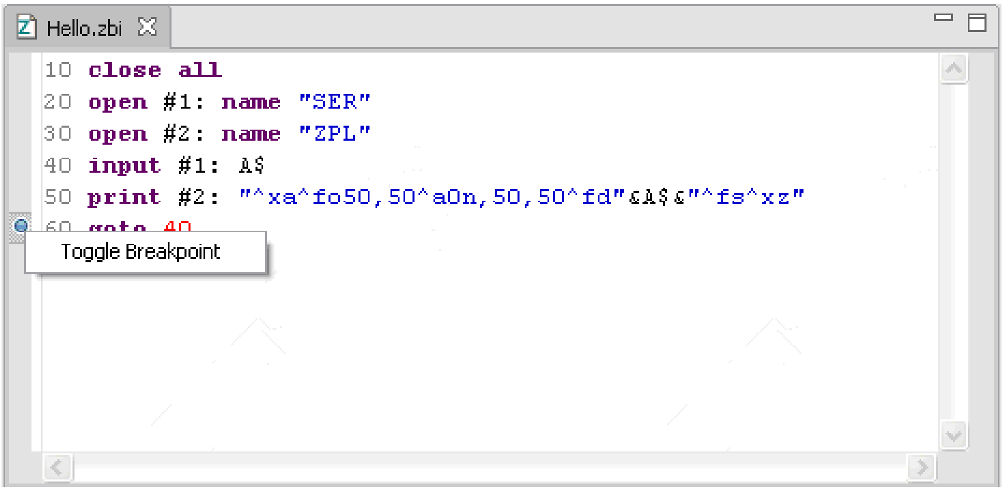
```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

Setting Breakpoints

A Breakpoint suspends the execution of a program at the location where the breakpoint is set. In the **ZBI Editor**, breakpoints can be enabled and disabled by using the displayed checkboxes. Double-clicking on a breakpoint in the **Breakpoints View** will highlight it in the **ZBI Editor**.

To set a breakpoint, right-click immediately to the left of a line of ZBI code and choose **Toggle Breakpoint**. In the case of the example program in this tutorial, put the Breakpoint on line 60. When the program is executed, the printer will stop each time it reaches that line.

Note • The maximum number of breakpoints per program is 16.



```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

Note • When a program containing breakpoints is sent to the printer, the breakpoints will also be transmitted–meaning that the program will stop whenever a breakpoint is reached. To avoid this situation, remove breakpoints before sending programs to printers. It is possible to remove all of the Breakpoints in the file by clicking on the ✖ icon in the Breakpoints View dialog.

Discovering Printers

Once a program is written and ready to be tested, it can be run on a printer – or run in the **Virtual Printer** environment.

If a physical printer is to be used, it will be necessary to first **Discover** and connect to a printer. **ZBI-Developer** supports using printers connected to a network using a ZebraNet print server. If a printer is connected via serial or parallel connection, a **Virtual Printer** should be used and data should be redirected to the port on the PC that the printer is connected to. (See *Using Virtual Printers on page 68* for more details.)

If the printer is connected to the same network subnet that the PC running ZBI-Developer is attached to, then a list of available printers will appear on the **Printer View** tab.

Once a printer has been discovered, it can be used to **Create a Debug Connection**, **Run/Debug a Program**, **Send Files to a Printer** or **Import Files From a Printer**.

Setting Up Searches

In some cases, the desired printer may not be automatically discovered. This can happen because the printer is on a different subnet, or because traffic on the network is being controlled in such a way that the discovery process is not allowed to complete. Many times, what is needed is a different type of search.

ZBI-Developer offers five types of network searches:

- Local Broadcast Search
- Subnet Search
- Directed Broadcast Search
- Multicast Search
- IP Address Search

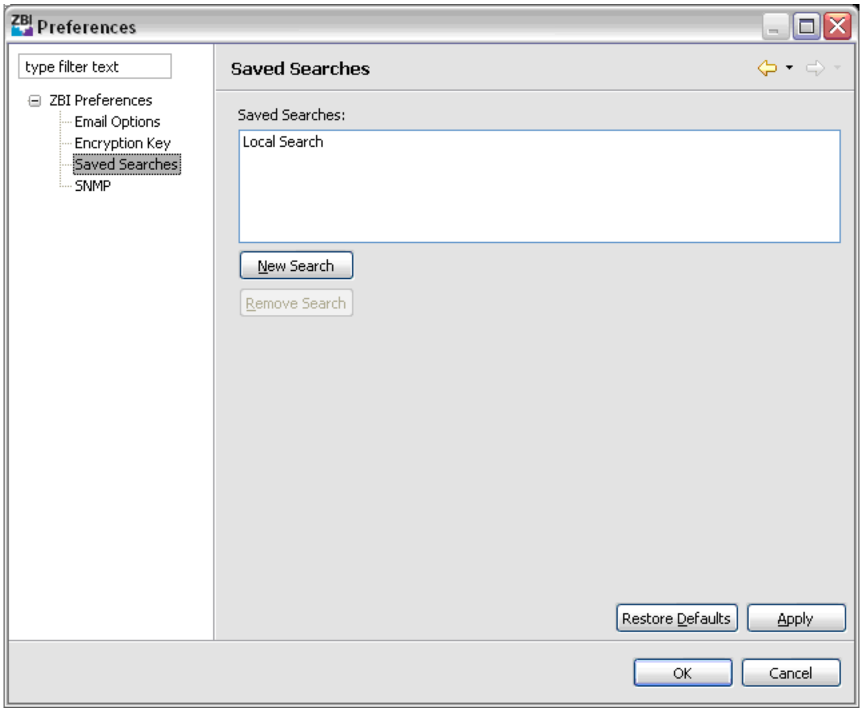
Note • The connection to a networked printer can be tested by right-clicking over the printer in the **Printer View** and choosing Print Configuration label. If the connection is active, a configuration label should print.

To set up a search for a network printer, do the following:

1. Go to the menu and choose **Window** > **Preferences**. The Preferences dialog will be displayed.

2. Click on the "+" symbol next to **ZBI-Preferences.**

3. Click on **Saved Searches**.



4. Click on **New Search**.

Local Broadcast Search

Local broadcasts send a discovery request to the local broadcast address of 255.255.255.255. This request finds printers in the same local network as the requesting computer. The local broadcast search is the default search type used by the program.

To create a local broadcast search, do the following:

1. In the Search Type dialog, select **Local Broadcast**. The Value fields are automatically populated.

2. Type a name in the **Search Group Name** textbox.

3. Click **Add**. The Value field is added to Search type.

4. Click **OK** to accept the new Search Type.

5. Click **OK** to exit the Preferences dialog.

Subnet Search

The Subnet Search sends a discovery request to each address in the specified subnet. This feature is useful for networks that prevent broadcast packets from passing across routers. The Subnet search requires you to type in the first three subnet octets.

Example: A request to 192.168.2.* sends a request to: 192.168.2.1 through 192.168.2.254.

To create a Subnet Search, do the following:

1. From the Search Type dialog, select **Subnet**.
2. Type a name in the **Search Group Name** textbox.
3. In the **Value** field, type the first three subnet octets.
4. Click **Add**.
5. Click **OK** to accept the new Search Type.
6. Click **OK** to exit the Preferences dialog.

Directed Broadcast Search

Directed broadcasts send a discovery request to the broadcast address of the specified subnet.

> **Note •** If a broadcast search does not work, the broadcast packets might be disabled across routers. Try Multicast or Subnet search.

> **Example •** The directed broadcast search requires you to type in the first three subnet octets, followed by 255, such as 192.168.2.255.

To create a Directed Broadcast search, do the following:

1. In the Search Type dialog, select **Directed Broadcast**. The fourth octet of the Value field is automatically populated.
2. In the **Value** field, type the first three subnet octets.
3. Type a name in the **Search Group Nam**e textbox.
4. Click **Add**.
5. Click **OK** to accept the new Search Type.
6. Click **OK** to exit the Preferences dialog.

Multicast Search

Multicast uses the multicast address to send a discovery address to all Zebra printers across as many routers as the hop count specifies. A multicast search may return a large number of printers in some installations.

This feature might be disabled on some routers and switches. See your network administrator to determine if a search failure was due to this feature being disabled.

To create a multicast search, do the following:

1. In the Search Type dialog, select **Multicast (Hops)**.
2. In the **Value** field, type the number of hops desired. The recommended starting point is 5
3. Type a name in the **Search Group Name** textbox.
4. Click **Add**.
5. Click **OK** to accept the new Search Type.
6. Click **OK** to exit the Preferences dialog.

IP Address Search

To create a search for an IP address, do the following:

1. In the Search Type dialog, select **IP Address xxx.xxx.xxx.xxx**.

   Example • Type in an IP address, such as 10.3.4.97.
2. In the Value field type in an IP address.
3. Type a name in the **Search Group Name** textbox.
4. Click **Add**.
5. Click **OK** to accept the new Search Type.
6. Click **OK** to exit the Preferences dialog.

Using a Printer via the Serial Port

If a printer is connected to the PC via the RS-232 Serial port, it is possible to use the Debugging features of ZBI-Developer, by making use of the Virtual Printer system.

Using a Printer via the Parallel Port

If a printer is connected to the PC via the Parallel Port, it is possible to use the Debugging features of ZBI-Developer, by making use of the Virtual Printer system.

Using Virtual Printers

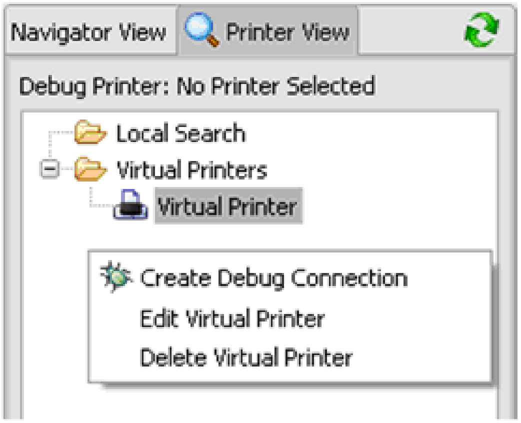
ZBI-Developer offers the ability to execute a ZBI program on the PC with no printer required. The Virtual Printer system makes it possible for the user to control the inputs and view the output of their program – all without having a printer attached to the computer.

Additionally, the Virtual Printer system makes it possible to connect devices to the PCs' RS-232 Serial or Parallel ports and direct input from those ports into their program. For example, if the ZBI program being written will need to accept data from coming into the printer via the RS-232 Serial port from a weigh scale, that event could be simulated by attaching the weigh scale to the PC and configuring the Virtual Printer to accept data from the computers serial port.

Next, if a printer is available, but it not yet ZBI-Enabled, the Virtual Printer system allows the output of the ZBI program to be sent to the computers parallel or serial ports to a printer. For example, a ZBI program could be written to send output to the printer label formatting engine – while the Virtual Printer is configured to take that same data and redirect it to the computers serial port to a printer attached to that port. In this way, the label formatting commands that are frequently the output of many ZBI program can be tested, even if a ZBI-Enabled printer is not immediately available.

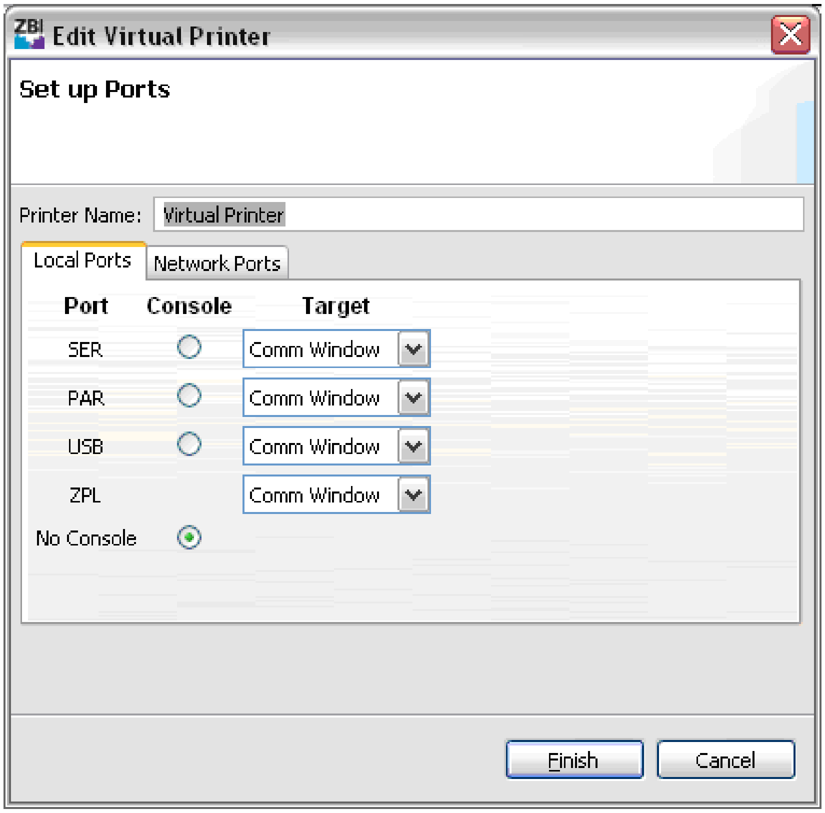
Edit a Virtual Printer

Virtual Printers can be edited to alter their settings. To Edit a **Virtual Printer,** right-click over the **Virtual Printer** in the **Printers View** and select **Edit Virtual Printer.**



Debug Printer: No Printer Selected

The **Edit Virtual Printer Wizard** will display.



Set up Ports

Printer Name: Virtual Printer

Port	Console	Target
SER	○	Comm Window
PAR	○	Comm Window
USB	○	Comm Window
ZPL		Comm Window
No Console	●	

Configure where output from the Virtual Printers ports should be directed. For this Tutorial, leave all of the Ports set to **Comm Window**.

> **Note •** Virtual Printers can be used to examine the output of a ZBI program. They do not have the ability to create an image of a printed label based on the receipt of printer commands. That capability is only present in the printer.

Creating a Debug Connection

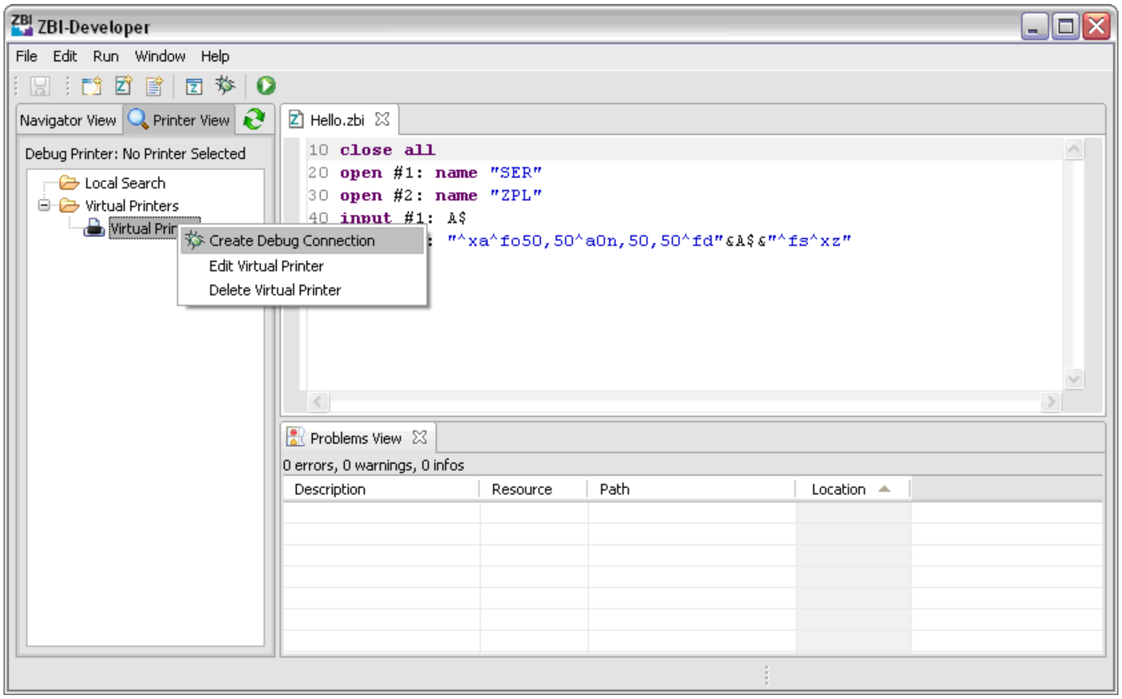
Once the printer to be used has been Discovered, or a Virtual Printer has been selected and configured, the next step is to run your program through the debugger environment.

ZBI-Developer offers several advanced features that allow the programmer to test the logic and flow of a program before it is deployed. Key features are the ability to:

- Set and view breakpoints
- View the data contained in variables as a program is running
- Input and view the data being sent to Ports defined in the ZBI program
- Pause, Run or Terminate the Debugging mode as needed.

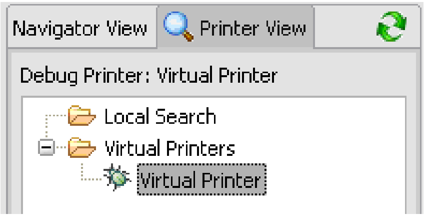
Once the printer type to be used has been determine (Virtual Printer or a networked printer) it is necessary to create a debug connection to the printer.

In all cases, Creating a Debug Connection is done in the following way – navigate to the **Printer View Tab**, right-click over the desired printer and choose **Create Debug Connection**. For the purposes of this tutorial, select a virtual printer.



```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
          : "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
```

After the debug connection has been made, the debug connection icon will appear next to the selected printer.



Note • Once a debug connection has been made to a printer, ZBI-Developer will attempt to maintain that connection, even if the program is shut down and re-started. Upon restarting, ZBI-Developer will attempt to reconnect to the selected printer.

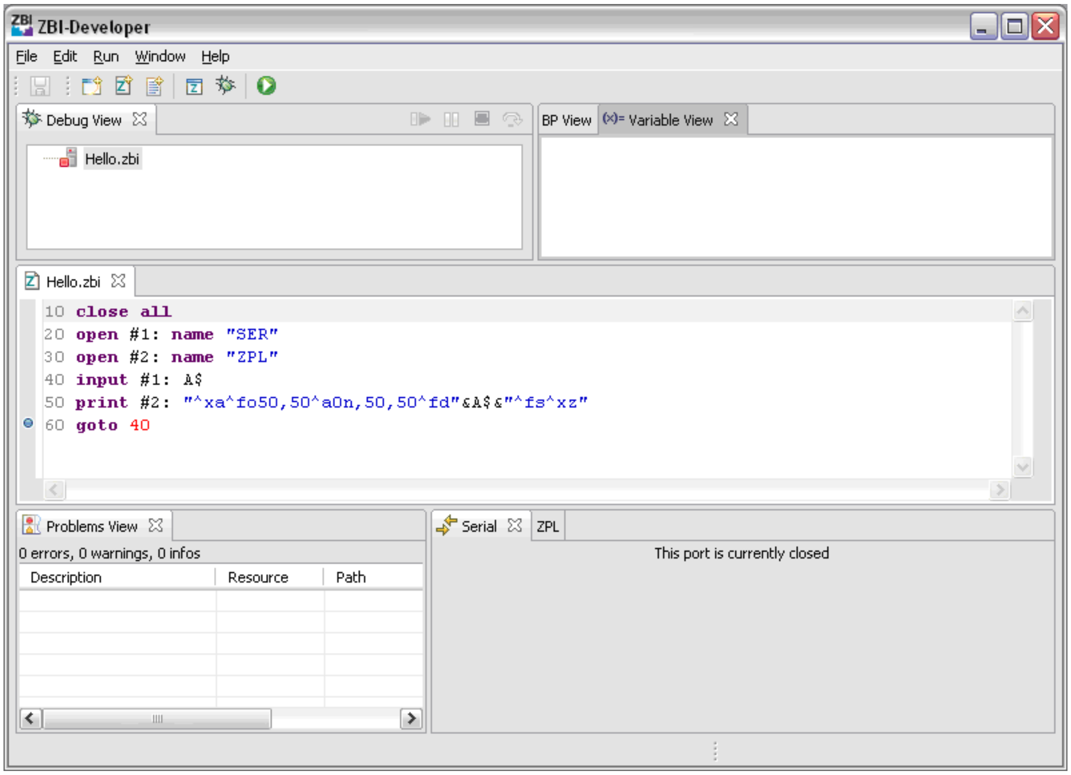
Note • Only ZBI-Enabled printers running firmware v60.16.x, v53.16.x or later support on-printer debugging. If you do not have a printer that meets this requirement, contact your reseller. For the purpose of this Tutorial, it is also possible to use a Virtual Printer if a ZBI-Enabled printer is not available, but a ZPL printer is.

Debug a Program

Once the necessary Breakpoints have been created and a Debug Connection has been made, the program can be run.

> **Note •** For the purposes of this Tutorial, the program will be run in the Virtual Printer environment, with output initially directed to the computer's screen instead of to a printer.

The first step is to change the Perspective from the **ZBI Perspective** to the **Debug Perspective**. Click on **Window > Show Perspective > Debug Perspective** or click the icon to change the perspective.



```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

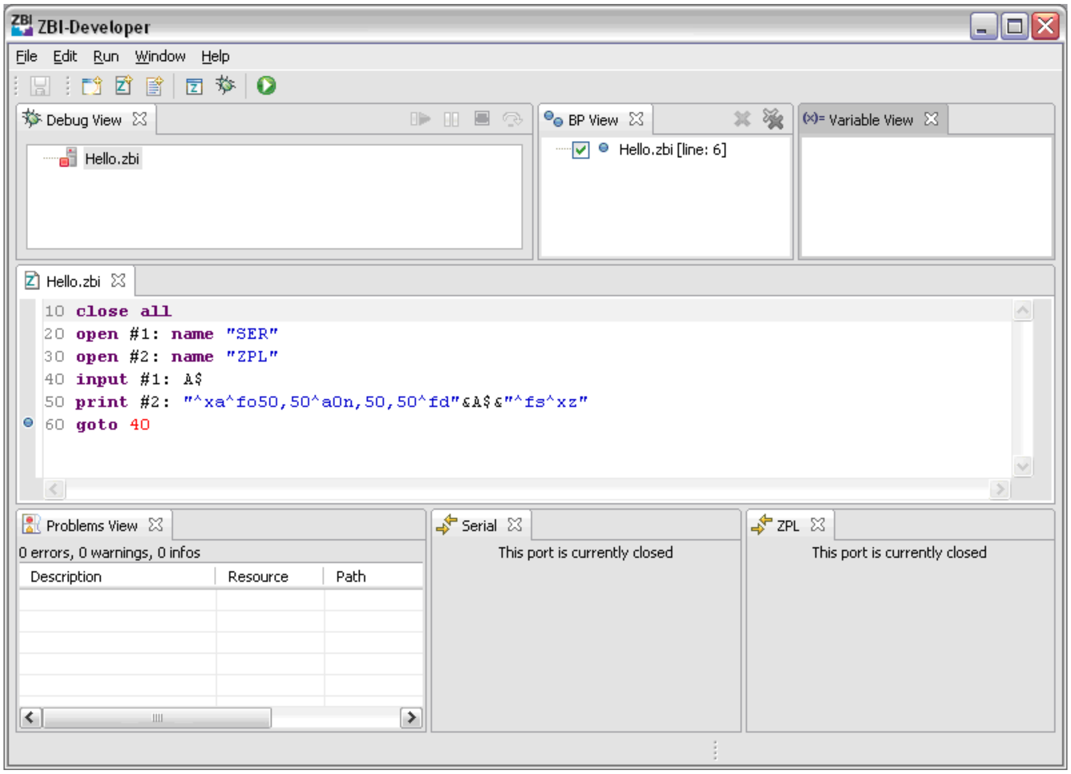
0 errors, 0 warnings, 0 infos

This port is currently closed

Changing the Screen Layout

The initial screen layout may be appropriate for some Debugging sessions, however the layout can also be changed to maximize the ability to see the various important Views that can be used while Debugging a program.

For this Tutorial, the Variable View, Serial Comm Window View and ZPL Comm Window View will be moved to make them all visible at the same time. To move a **View**, click and drag it to a new location. While a View is being moved, an outline of the View will be visible. Rearrange the Debug Perspective so that it looks like this:



☑ Hello.zbi [line: 6]

```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

0 errors, 0 warnings, 0 infos

Description	Resource	Path

This port is currently closed

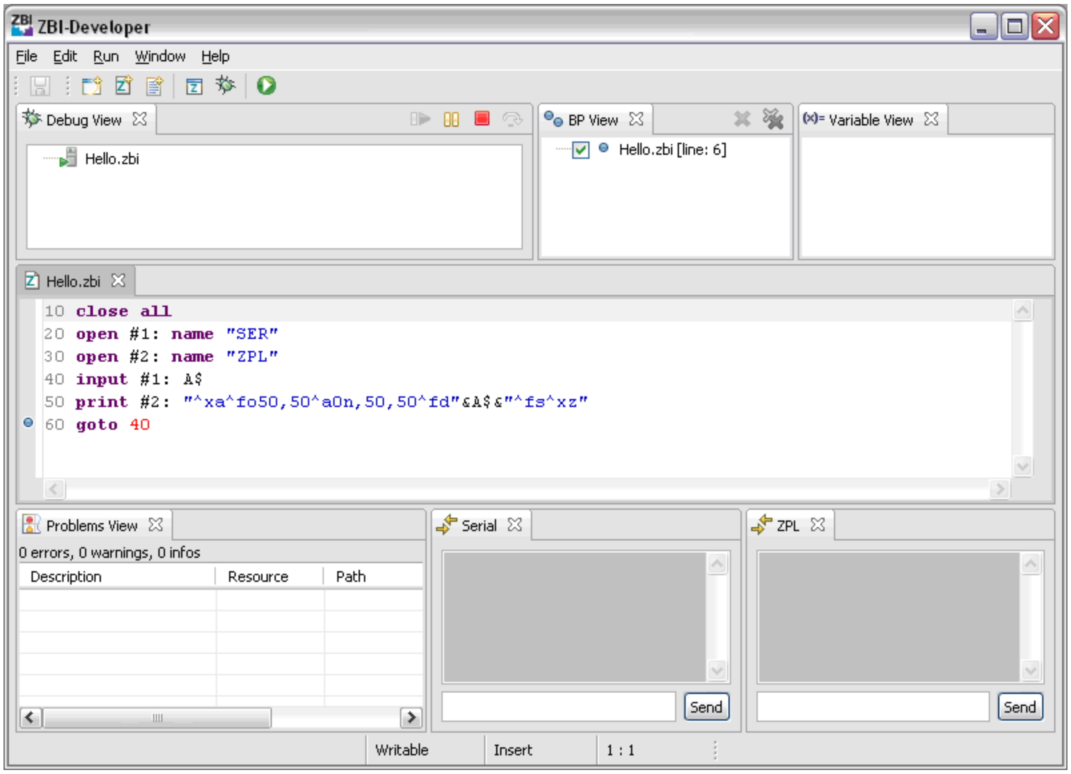
This port is currently closed

Run a Program

To run the program you’ve just written, click on menu **Run > Run/Debug Program** or click the ▶ icon.

Notice that several things have changed on the screen.

- The program has executed to line 40 and is now awaiting input.
- A data entry prompt is now displayed in the **Serial** and **ZPL Views**
- The icons in the Debug View have changed so that the **Terminate** 🟥 and **Pause** ⏸ icons are now active. **Do not click these yet!**

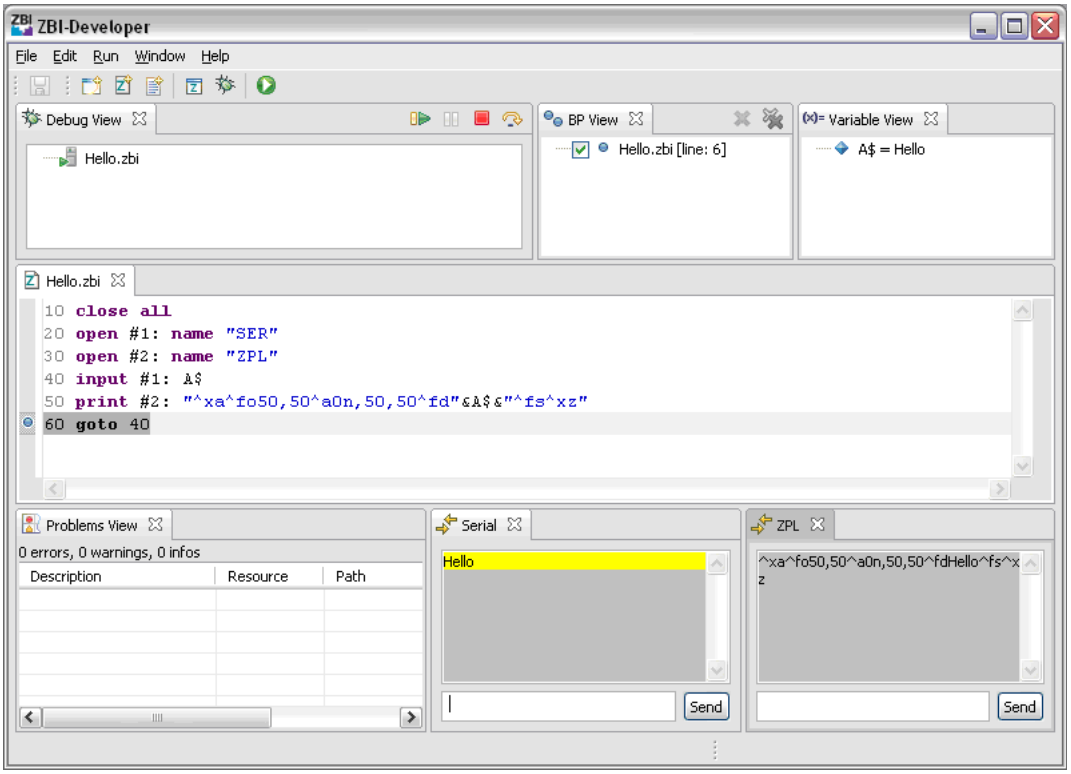


```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

To test the program, type "Hello" into the data entry prompt in the **Serial View** and click **Send**; this simulates data coming into the printer via the RS-232 serial port.

Notice the changes on the screen.

- The A$ variable has been filled with the data you sent via the serial port.
- The **ZPL Port View** has been filled with the ZPL command from Line 50 of the program. This is the output of the program. If the program was being run on a printer, or if the Virtual Printer was configured to direct the ZPL out to a printer connected to the PC, this is what the printer would use to print a label.
- The **Resume** and **Step Over** icons are now active in the **Debug View**
- Line 60 of the program is Highlighted, showing the line the program has stopped on – because of the Breakpoint.



☑ Hello.zbi [line: 6]

A$ = Hello

```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

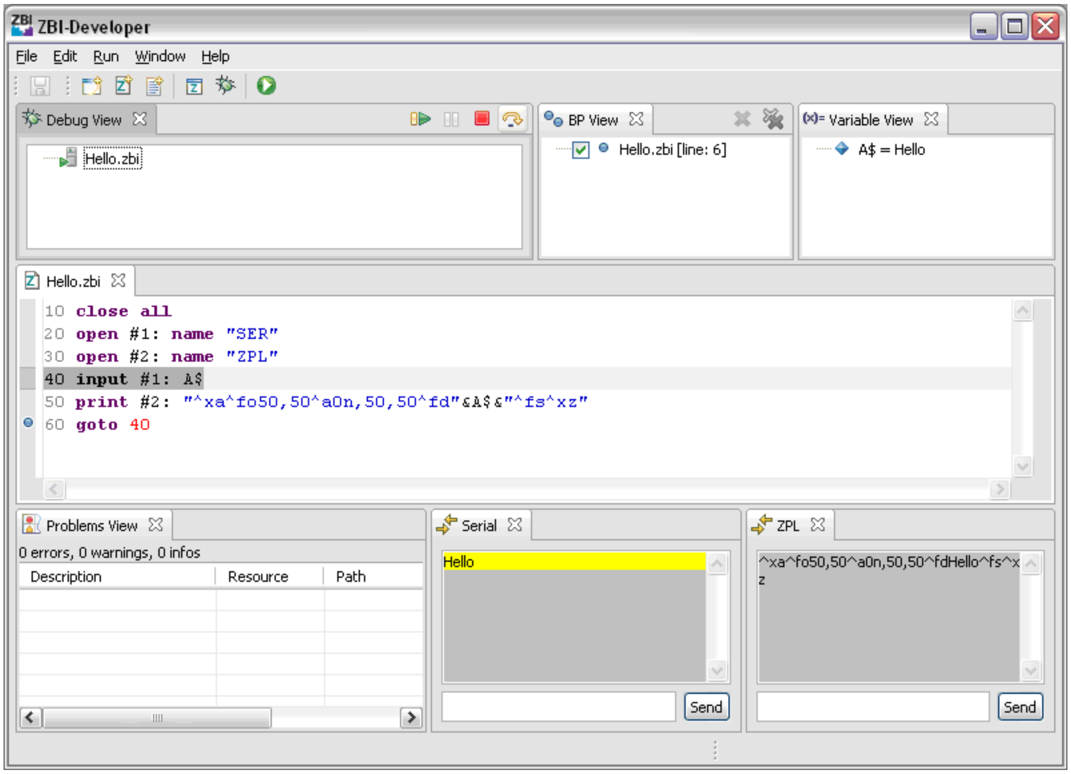
0 errors, 0 warnings, 0 infos

```
Hello
```

```
^xa^fo50,50^a0n,50,50^fdHello^fs^x
z
```

Step Over a Breakpoint

Now that the program is at a Breakpoint, you can Step Over the breakpoint and allow the program to execute to its next logical step. Click on the **Step Over** icon and note that program has executed to Line 40, again looking for input into the serial port.



Hello.zbi [line: 6]

A$ = Hello

```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

0 errors, 0 warnings, 0 infos

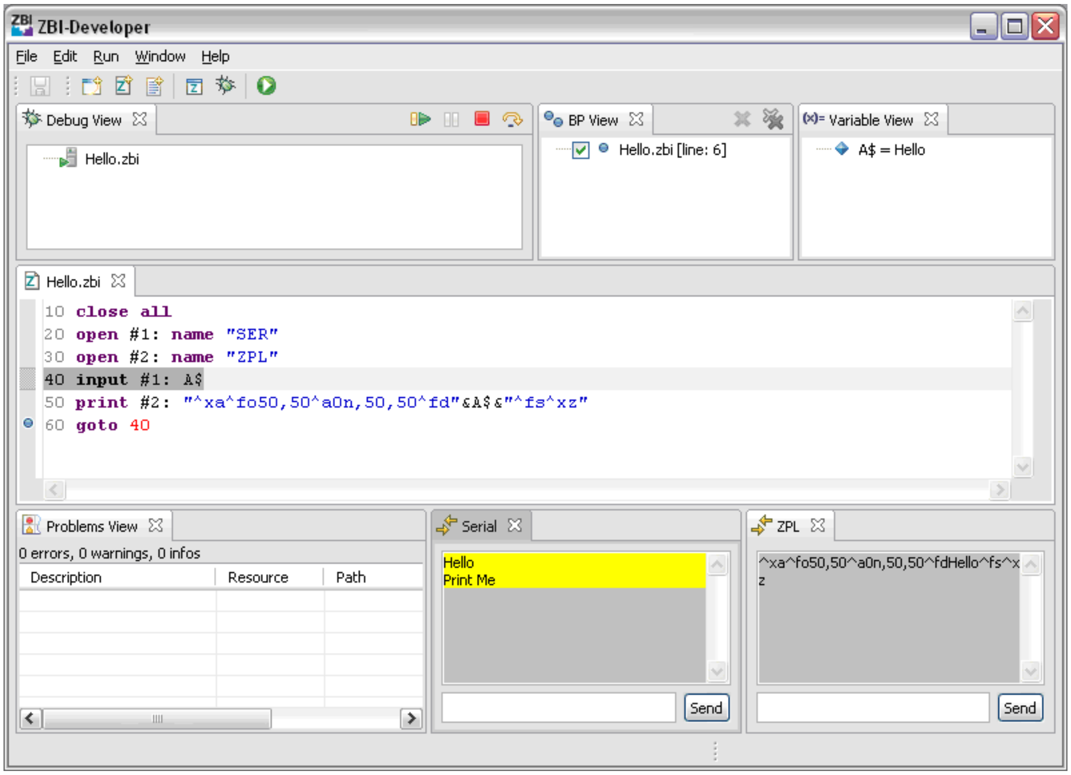
Description	Resource	Path

```
Hello
```

```
^xa^fo50,50^a0n,50,50^fdHello^fs^x
z
```

Type "Print Me" in the data entry prompt in the **Serial Port View** and click **Send**; this simulates data coming into the printer via the RS-232 serial port.

> **Note •** The program is still on Line 40.



Debug View: Hello.zbi

BP View: ☑ Hello.zbi [line: 6]

Variable View: A$ = Hello

```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

Problems View: 0 errors, 0 warnings, 0 infos

Description	Resource	Path

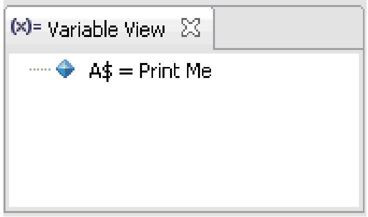
Serial:

```
Hello
Print Me
```

ZPL:

```
^xa^fo50,50^a0n,50,50^fdHello^fs^x
z
```

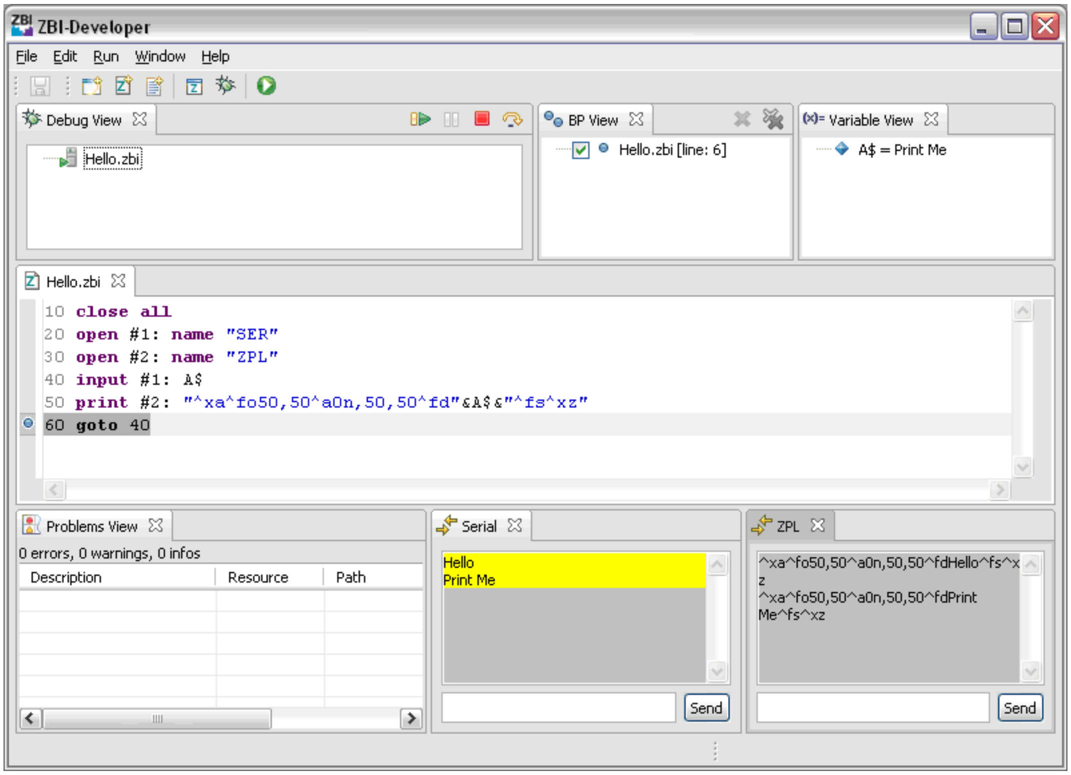
Click the **Step Over** icon again to move the new data into the **Variable View.**



A$ = Print Me

Next, click the **Step Over** icon again to move the program to the next step – embedding "Print Me" into the ZPL defined on Line 50.

> **Note •** A new line of ZPL is shown in the **ZPL Port View**.



Hello.zbi

Hello.zbi [line: 6]

A$ = Print Me

```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

0 errors, 0 warnings, 0 infos

Description	Resource	Path

```
Hello
Print Me
```

```
^xa^fo50,50^a0n,50,50^fdHello^fs^x
z
^xa^fo50,50^a0n,50,50^fdPrint
Me^fs^xz
```

Terminating a Running Program

To stop a running program, click on the **Terminate** 🟥 icon in the **Debug View.** Any active **Port Views** will be deactivated and defaulted. The last value stored in the variable will remain in the **Variables View**. The Step Over, Terminate and Resume icons in the Debug View will be deactivated. If the program was being debugged on a printer instead of in a Virtual Printer, it will be stopped.

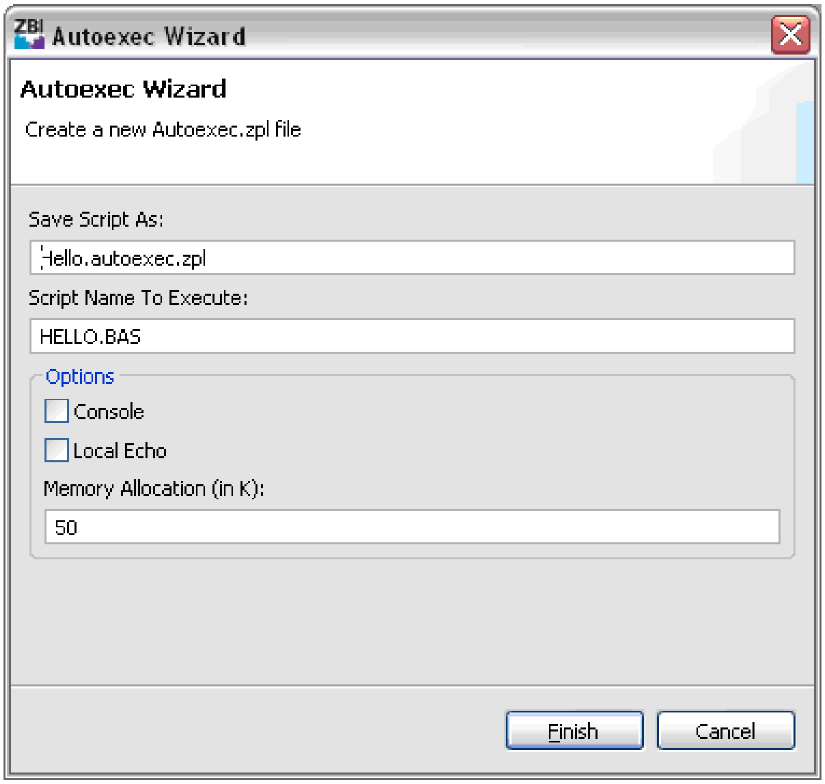
Creating the Autoexec.zpl file

ZBI-Developer can automatically create an Autoexec.zpl file for use with the program being created.

An Autoexec.zpl file is used to cause the printer to automatically run a set of commands when the printer is first powered up. In the context of a ZBI program, the Autoexec.zpl file can cause the program to be executed when the printer is turned on.

> **Note •** ZBI-Developer creates an Autoexec.zpl file that addresses only the need to start a selected ZBI program. If the user requires that the Autoexec.zpl file perform additional functions, the file will require editing to include those functions.

To create an Autoexec.zpl file, right-click over a .zbi file in a Project and select **Generate Autoexec.zpl**. The Autoexec Wizard will display. Name the file and select the **Console** and **Local Echo** options as needed. Consult the ZPL manual for details on how to use these options.



Autoexec Wizard

Create a new Autoexec.zpl file

Save Script As: Hello.autoexec.zpl

Script Name To Execute: HELLO.BAS

Options

□ Console

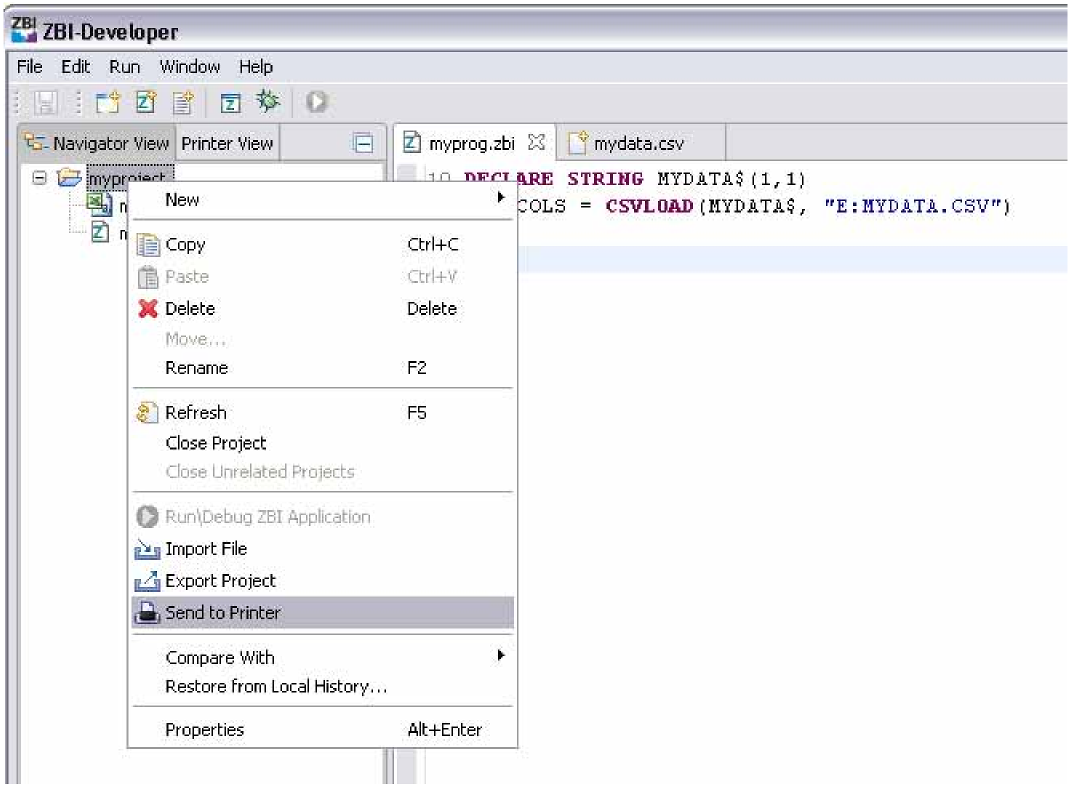
□ Local Echo

Memory Allocation (in K): 50

Sending Files to Printers

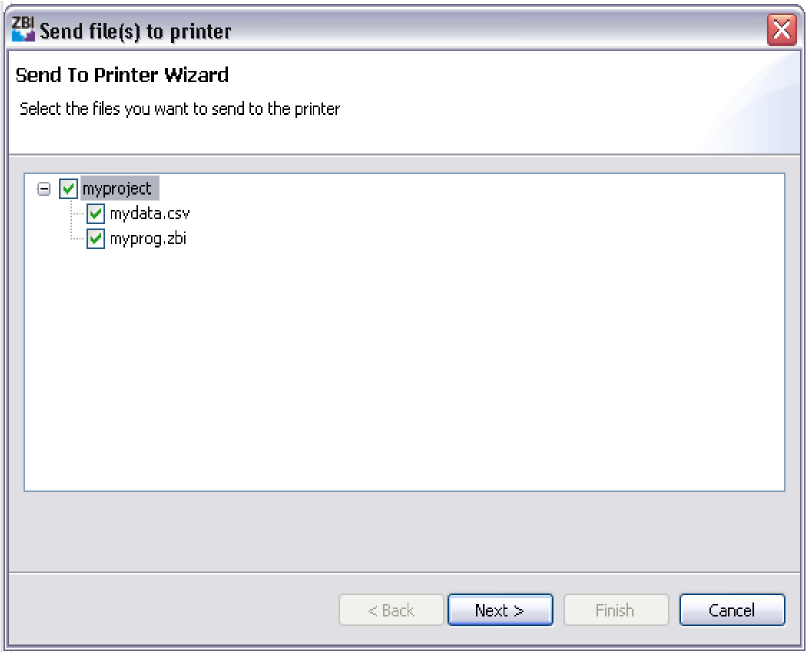
When the software has been connected to a printer, ZBI-Developer can send individual files or entire projects to one or more printers, depending upon the connection type. ZBI-Developer can send files to a debug printer, one or more networked printers, or a printer connected via parallel or serial port.

To send a file or project to a connected printer, right-click on a file or project and choose **Send to Printer**. The Send to Printer wizard will display.



```
10 DECLARE STRING MYDATA$(1,1)
   COLS = CSVLOAD(MYDATA$, "E:MYDATA.CSV")
```

If a project was selected, the wizard starts with a step to choose which files should be sent to the printer(s).



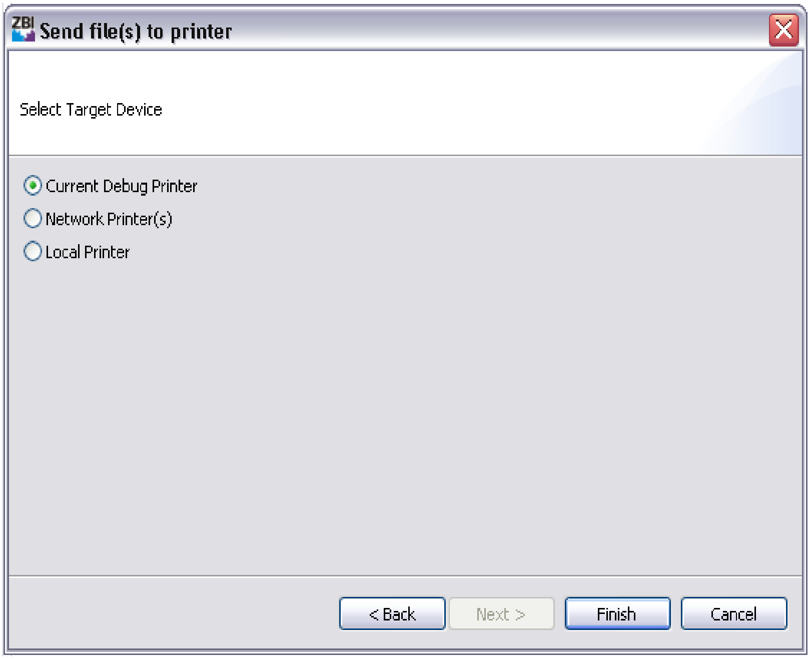
Send file(s) to printer

Send To Printer Wizard

Select the files you want to send to the printer

- ☑ myproject
  - ☑ mydata.csv
  - ☑ myprog.zbi

After pressing next on the first wizard step for sending a project, or if a single file is selected from the right-click menu, the wizard displays the available target devices.



Send file(s) to printer

Select Target Device

● Current Debug Printer
○ Network Printer(s)
○ Local Printer

Send File to Current Debug Printer

Select the **Current Debug Printer** option and click on **Finish** to send the selected project or file to the current debug printer.

Send File to Network Printer(s)

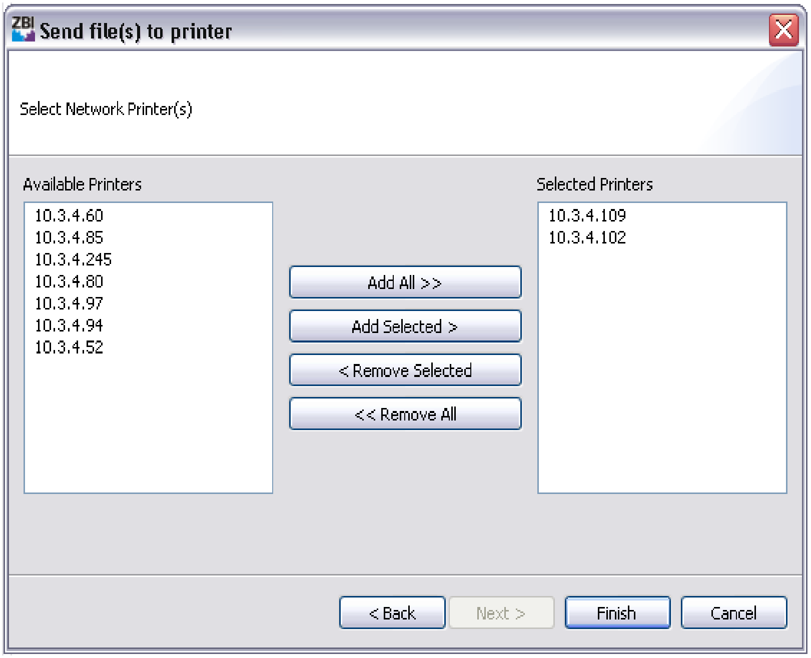
Select the **Network Printer(s)** option and click next to move to the next step in the Send to Printer Wizard.

The list of printers in the **Available Printers** column is a combination of all printers that are currently listed in the **Printer View** tab in the ZBI perspective (See *Discovering Printers on page 64* for more details).

To send the project or file to all printers in the **Available Printers** list click the **Add All** button. This will move all available printers to the **Selected Printers** column.

To send one or a select list of printers, individually select each printer in the **Available Printers** column and click on **Add Selected** to move the selected printers to the **Selected Printers** column. Hold down the CTRL key while clicking on individual items to select more than one printer from the **Available Printers** column.

Click **Finish** when the desired printers are in the **Selected Printers** column.



Send file(s) to printer

Select Network Printer(s)

Available Printers
10.3.4.60
10.3.4.85
10.3.4.245
10.3.4.80
10.3.4.97
10.3.4.94
10.3.4.52

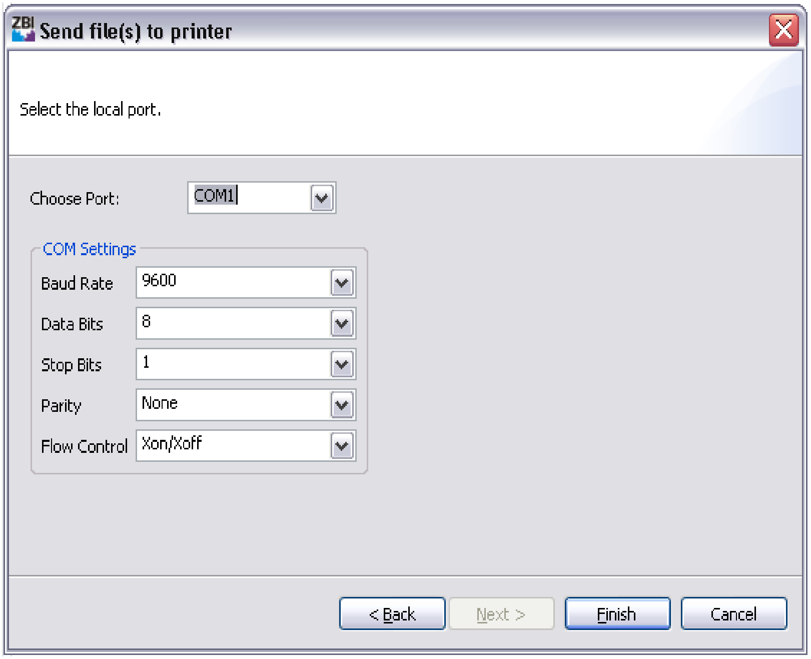
Selected Printers
10.3.4.109
10.3.4.102

Send File to Local Printer

Select the **Local Printer** option and click **Next** to move to the next step in the Send to Printer Wizard.

The **Choose Port** drop down will list all local ports that are available. Click **Finish** to send the project or file to the local port.

If a serial port is selected in the **Choose Port** drop-down, the serial port settings are displayed. The settings should be set to match the settings on the connected printer.



Send file(s) to printer

Select the local port.

Choose Port: COM1

COM Settings

Baud Rate: 9600

Data Bits: 8

Stop Bits: 1

Parity: None

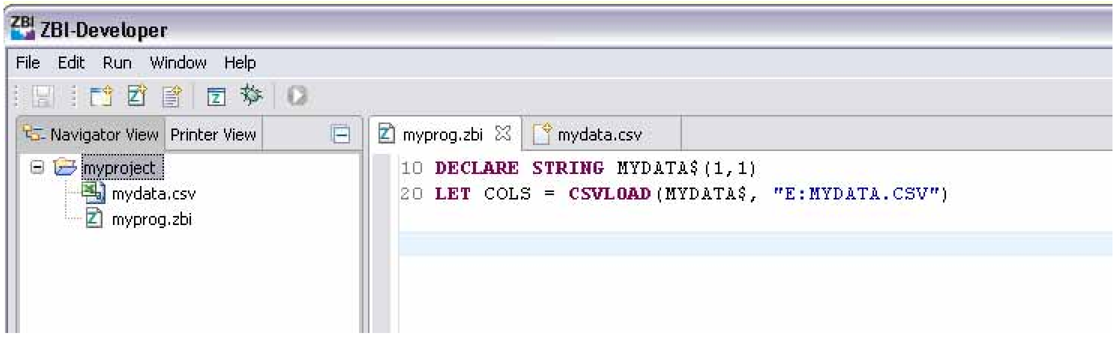
Flow Control: Xon/Xoff

Using CSV/TXT Files with Virtual Printers

If a CSV or TXT file is added to the project (See *Project Files on page 55*), it can be accessed (e.g., CSVLOAD and TXTLOAD) via the Virtual Printer. Likewise, stored files (e.g., CSVSTORE and TXTSTORE) will be stored in the project directory. To do so, simply treat the file as if it were on the printer using the DRIVE:FILENAME.EXT format.

Example • If there is a file named MYDATA.CSV in the project (see figure below) it can be loaded via the CSVLOAD command as follows:

```
20 LET COLS = CSVLOAD(MYDATA$, "E:MYDATA.CSV")
```



```
10 DECLARE STRING MYDATA$(1,1)
20 LET COLS = CSVLOAD(MYDATA$, "E:MYDATA.CSV")
```

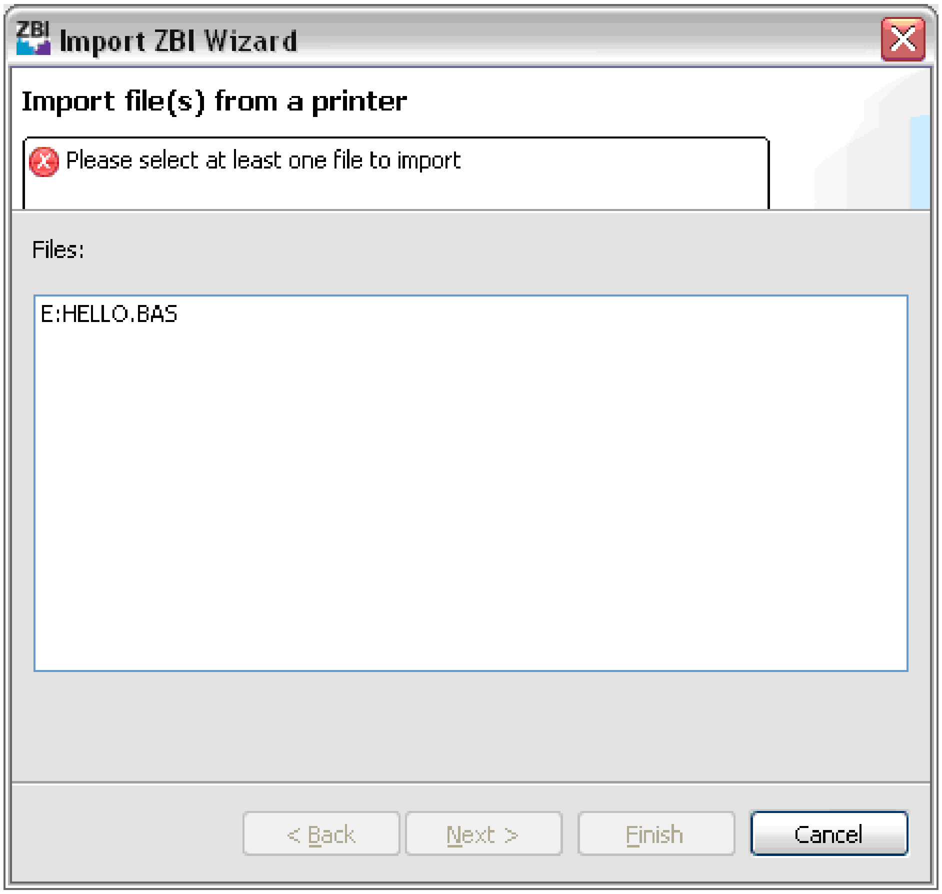
The above example will load the mydata.csv file located in the project 'myproject'. The file names are not case sensitive.

Do not specify the full Windows PC path name for the file. The Virtual Printer will replace any of the Zebra Printer Drives (Z, E, B, R, and A) with the path of the project. If the CSVfile is in the root of the project in the Navigator View, then the Virtual Printer will be able to find it. Refreshing the project contents will show any files that were created by a program that calls CSVSTORE or TXTSTORE.

Import ZBI Files From a Printer

When the software has been connected to a networked printer, ZBI-Developer can retrieve ZBI programs from a printer. See the information on *Creating a Debug Connection on page 70* for details on how to connect to a printer.

To retrieve ZBI programs from a printer, right-click over the printer and select **Import File**.



Import file(s) from a printer

Please select at least one file to import

Files:

E:HELLO.BAS

The Import Wizard will display. Select the file(s) to import from the printer to the PC and then the Project to import the file(s) into. Click **Finish** to complete the Wizard.

> **Note •** Encrypted ZBI programs (with a .bae extension) cannot be imported from the printer back to ZBI-Developer. It is recommended that you archive Encrypted ZBI programs.

Compare or Replace Files

ZBI-Developer maintains a log of file changes, making it possible to compare or replace the current version of a file with prior versions. This feature can be very useful when iterative changes are being made to a program.

To use the **Compare** feature, right-click over a file and select **Compare With > Local History.** The Compare with Local History dialog will be displayed. If a file has gone through multiple changes and saves, each of those event will be available to select for comparison with the current version. By selecting **Replace With > Local Version** it is possible to choose an earlier version of the file to replace the current version.

Additionally, two files can be compared with each other. To compare two files, hold down the **Ctrl** on the keyboard and select two files. Then right-click over one of the files and choose **Compare With > Each Other**.

Encryption of ZBI Programs

This section details how to optionally encrypt programs before distributing them.

> **Note •** It is recommended that Projects and Project files be backed up on a regular basis. Zebra Technologies and ZBI-Developer do not automatically back-up files or offer a method for un-encrypting programs.

ZBI-Developer can encrypt ZBI programs for use on printers that support ZBI 2.0 and above. The purpose of this feature is to allow the programmer to protect the source code of the program once it has been deployed to printers. Programs that have been encrypted are represented in the ZBI-Developer environment with the "encrypted ZBI program" icon instead of the standard ZBI program icon .

> **Note •** The Encryption Key and the source code for ZBI programs should be regularly archived.

While the encrypted program will remain in plain text in the ZBI-Developer environment, it will be encrypted and MIME encoded when it is sent to the printer.

The Default Encryption Key

ZBI-Developer has a unique encryption key that is created when the program is installed. This key is the default key used when a ZBI program is encrypted – unless another key has been created.

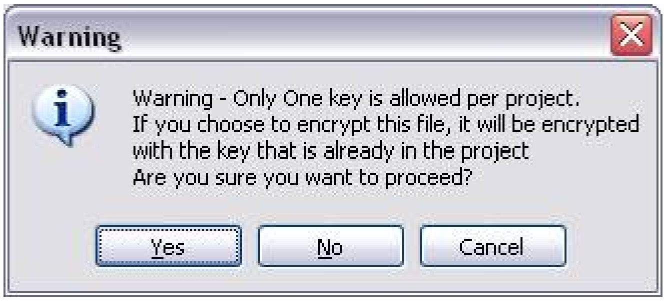
> **Note •** It is recommended that the Default Encryption Key be used rather than creating additional keys. This approach is recommended to make the task of managing the keys easier over time.

Encryption keys can be shared between users in one of two ways:

- The key file can be copied from one machine to another. This approach is recommended when program development and encryption will be done from remote locations.
- The key file can be stored on a networked drive, made available to any of the programmers to use.

Encrypt a ZBI Program

To encrypt a ZBI program, right-click over the program and choose **Encrypt ZBI Program**. The program will display a Warning dialog as shown below:

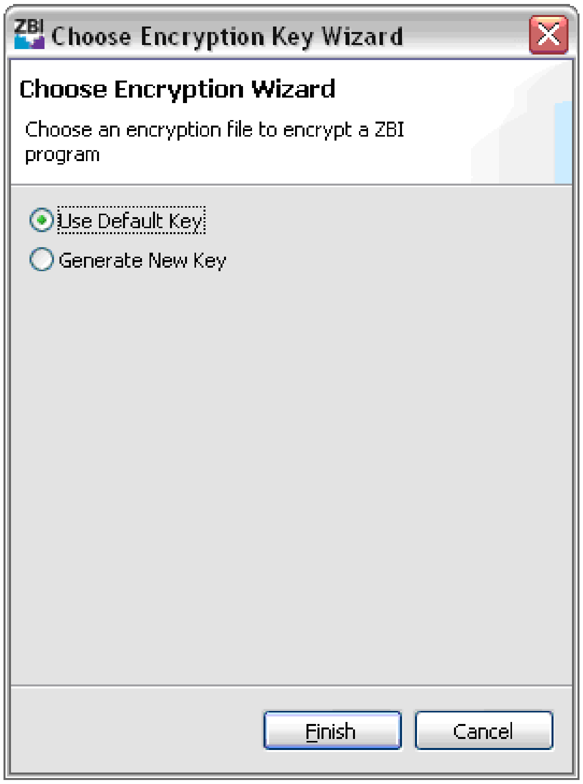

Warning

Warning - Only One key is allowed per project.
If you choose to encrypt this file, it will be encrypted
with the key that is already in the project
Are you sure you want to proceed?

Next, the program will offer the choice to use the **Default Encryption Key** or to **Generate a New Key**.

> **Note** • It is recommended that the Default Encryption Key be used rather than creating additional keys. This approach is recommended to make the task of managing the keys easier over time.

Choose which Key to use and click **Finish** to complete the Encryption Key Wizard. The ZBI program will be marked as encrypted. When it is sent to a ZBI 2.0 printer, it will be encrypted and MIME encoded.


Choose Encryption Key Wizard

Choose Encryption Wizard

Choose an encryption file to encrypt a ZBI program

● Use Default Key
○ Generate New Key

Example • The program from the ZBI-Developer tutorial is shown below in plain text form:

```
10 close all
20 open #1: name "SER"
30 open #2: name "ZPL"
40 input #1: A$
50 print #2: "^xa^fo50,50^a0n,50,50^fd"&A$&"^fs^xz"
60 goto 40
```

When encrypted and sent to the printer, the program is transmitted using the Base64 encoding scheme. (To improve readability, line breaks have been added.)

```
^XA^IDE:HELLO.BAS^XZ
^XA^IDE:HELLO.BAE^XZ
^XA^IDR:HELLO.BAS^XZ
^XA^IDR:HELLO.BAE^XZ
~DZE:HELLO.BAE,176,:B64:SEVMTE8AAAAAACUAAAAAoPDiWDoW22
sG86Nnf7HUfLZB3YSMil9HwPcpzIMs4buW+FDGw2FY89osEEofNF9e
uU8dz5872jW8Gafq85vsr47SydHGIjymIVxQq6c50GRBJjFrEVkh+u
vhnHWQ7HRasXvB7N/VfB0qKSDiICdwzNkRZL5G3nFAQFIQtfDyrv51
IBYfGRn+4lxHcPRUGNveleNcSP58qjt+UBXziVVLlv0=:61b3
```

Distributing Encrypted Programs

When an Encrypted ZBI 2.0 program is distributed to a printer, they program will appear on the printers directory listing page with the extension "`.bae`" (basic encrypted). The file will be listed in on the printer's directory listing page, but it will not be possible to return the program to the host in plain text format.

Note • The program will not run unless the encryption key has been sent to the printer. To send the encryption key to the printer, right click over the key and click "Send to Printer". The key can also be exported out of ZBI-Developer to a file on the computers hard drive – and then that exported file can be sent to printers as needed. Zebra does recommend maintaining control over access to the Encryption key in order to benefit from the protection the Encryption system offers.

Generate a New Key

The default encryption key is the recommended key to use, however it may be desirable to create a new key. For example, a developer may want to create a new key for a unique project. A new encryption key can be created when the user chooses to encrypt the program.

To create a new key, select **Generate New Key** when encrypting a program. A new unique key will be created and displayed in the Project folder with the Key 🔑 icon.

Key Storage

Note • It is recommended that Projects and Project files be backed up on a regular basis. ZBI-Developer does not automatically back-up files or offer a method for un-encrypting programs if the Encryption Key is not present.

ZBI File Properties

The properties for each ZBI file can be viewed and set. To set the file properties, right-click over a ZBI file in the **Navigator View** and choose **Properties**. The file size, location, encoding type and other settings are displayed on the Info dialog. Additionally, the file attributes can be set on the **ZBI Options Info Page**.

Note • Setting the attributes for a ZBI program will change how it is presented on the printers web pages. If a file is configured to be persistent, it can only be deleted if it is specifically named in a command. For example, if a persistent file is named "`hello.bae`" was stored in E memory, the ZPL command to delete it would be "`^XA^IDE:Hello.bae^FS^XZ`".

ZBI Technical Support

Online Support

You can find the latest builds and updates under the Support section on the Zebra Web site.

If you cannot resolve the issue, please contact your local reseller or the offices listed in *Contact Information***.**

Contact Information

Worldwide, Technical Support is available through your Reseller, or at: http://www.zebra.com/support.

Professional ZBI Programming services can be obtained by contacting:

zbi-experts@zebra.com

Service Quotations available upon request.

Zebra Technologies Corporation
International Headquarters
333 Corporate Woods Parkway
Vernon Hills, Illinois 60061-3109 USA

Phone: +1 847 913 2259
Fax: +1 847 913 2578

Notes • ____

Index

Zebra Technologies International, LLC
333 Corporate Woods Parkway
Vernon Hills, Illinois 60061.3109 U.S.A
T: +1 847 793 2600
Toll-free +1 800 423 0422
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

http://www.zebra.com