ProjectPrinter Suite - Comprehensive Manual

1. Introduction

ProjectPrinter is a professional-grade virtual line printer emulator designed for mainframe and minicomputer enthusiasts. It acts as a bridge between your simulated hardware (Hercules, SIMH, etc.) and modern output formats, generating authentic-looking Greenbar PDF reports or plain text files.

This repository contains three distinct projects that work together to provide a complete printing solution:

1. **ProjectPrinter**: The core service that listens for print jobs, processes the data (including EBCDIC/ASCII conversion and overstrike handling), and generates the output.
2. **device_config**: A local console-based utility (TUI) for managing your printer configurations directly on the host machine.
3. **Device_Config3270**: A unique configuration server that allows you to manage your printers using a TN3270 terminal emulator, providing an authentic mainframe-style experience.

---

2. Installation and Setup

Prerequisites

ProjectPrinter is built on **.NET 9.0**. You must have the .NET 9.0 Runtime installed to run the applications, or the .NET 9.0 SDK to build them from source.

- Download .NET 9.0

Cloning the Repository

```
git clone https://github.com/SComps/ProjectPrinter.git
cd ProjectPrinter
```

Building the Projects

You can build the entire solution using Visual Studio 2022+ or the `.NET CLI`.

Using CLI:

```
# Publish the core printer service (AOT)
dotnet publish ProjectPrinter/ProjectPrinter.vbproj --self-contained -c Release -o

# Publish the local config utility (AOT)
dotnet publish device_config/device_config.vbproj --self-contained -c Release -o .

# Publish the TN3270 config server (AOT)
dotnet publish Device_Config3270/Device_Config3270.vbproj --self-contained -c Rele
```

The compiled, self-contained executables will be located in the specified output directories (e.g., `publish/ProjectPrinter/`).

---

3. Configuration

All three applications in the suite share a single configuration file, typically named `devices.dat`. This file defines the virtual printers, their connection details, and output preferences.

You can edit this file manually, but it is highly recommended to use one of the two comprised configuration tools.

Option A: Local Console Configuration (`device_config`)

This tool runs in your terminal and provides a simple menu-driven interface.

How to Run:

```
cd device_config
dotnet run
# OR
./device_config [path_to_config_file]
```

(Default config file: `devices.dat` in the current directory)

Usage:

- **Navigation**: Use `PageUp` / `PageDown` or type `UP` / `DOWN` to scroll through devices.
- **Add Device**: Type `ADD`.
- **Edit Device**: Type the **ID number** of the device (e.g., `1`, `2`).

- **Delete Device**: Type `DELETE <ID>` (e.g., `DELETE 1`).
- **Save Changes**: Type `SAVE`.
- **Exit**: Type `EXIT`.

Editing Fields:
When editing a device, use `Enter` to confirm a field and move to the next. Use the arrow keys to navigate within a text field.

Option B: TN3270 Remote Configuration (`Device_Config3270`)

For the true mainframe enthusiast, this tool runs a mini TN3270 server. You can connect to it using your favorite 3270 emulator (x3270, wc3270, Tom Brennan's Vista, etc.) and configure your printers using a CICS-like interface.

How to Run:

```
cd Device_Config3270
dotnet run -- -p 3270
```

- `-p <port>`: Specifies the listening port (default is 3270).
- You can also specify the config file path as a second argument.

Connecting:
Open your TN3270 emulator and connect to `localhost:3270` (or the IP of the machine running the tool).

Usage:

The interface mimics a mainframe transaction screen.

- **PF7 / PF8**: Page Up / Page Down.
- **PF3**: Exit / Back.
- **Command Line**: Type `ADD`, `SAVE`, or the ID of a device to edit.
- **Fields**: Tab between fields and type naturally.

Configuration Fields Explained

When configuring a printer (using either tool), you will see the following fields:

Field	Description
Device Name	A friendly name for the printer (e.g., `MVS_PRT0`).

Description	A brief description (e.g., `Job Output Printer`).
Device Type	`0` = Printer, `1` = Reader (Reader support is experimental).
Connection Type	`0` = Socket (Most common), `1` = File, `2` = Physical.
Destination	The IP address and port of the host system to connect to (e.g., `127.0.0.1:1403`).
OS	The source OS. Critical for correct carriage control handling. 0: MVS 3.8J 1: VMS 2: MPE (HP3000) 3: RSTS/E 4: VM/370 5: NOS 2.7.8 6: VM/SP 7: Tandy XENIX 8: z/OS
Auto Connect	`True`: Start listening immediately inside ProjectPrinter. `False`: Requires manual start (rarely used).
Output PDF	`True`: Generate a Greenbar PDF. `False`: Generate plain text only.
Orientation	`0`: Landscape `1`: Portrait `2`: Landscape (No Background) `3`: Portrait (No Background)
Output Dir	The directory where PDFs/Text files will be saved. Can be relative or absolute.

---

4. Running the Virtual Printer (`ProjectPrinter`)

Once your devices are configured in `devices.dat`, you are ready to start the main engine.

Command Line Syntax:

```
./ProjectPrinter [options]
```

Options:

- `config:<filename>` : Specify a custom config file (Default: `devices.dat`).
- `logType:<type>` : Control logging.
  - `printers.log` (Default): Log to a file named printers.log.
  - `default` : Log to the console window (stdout).
  - `none` : Disable logging.
  - `<filename>` : Log to a specific custom file.
- `--imageproc` : Force the use of the legacy image-based background processor (slower, but provided for compatibility).
- `test` : Runs a diagnostic test that generates sample greenbar PDFs in the current directory.
- `version` : Displays the current version and exits.

Example:

Run with console logging (useful for debugging):

```
./ProjectPrinter logType:default
```

Run as a background "service" (logging to file):

```
./ProjectPrinter
```

> **Hot-Reloading:** ProjectPrinter monitors the `devices.dat` file. If you make changes using `device_config` or `Device_Config3270` while the printer engine is running, it will automatically detect the changes, disconnect affected devices, and reload the configuration without needing a restart.

---

5. Connecting Your Emulator

To use ProjectPrinter, configure your emulator (Hercules, SIMH, etc.) to connect to the socket defined in your device configuration.

Example: Hercules (MVS/VM)

If you configured a printer in ProjectPrinter with **Destination** `1403` and using the host `127.0.0.1`:

In `hercules.conf`:

```
# Device  Type  Address
000E      1403  127.0.0.1:1403 sockdev
```

Note: Ensure the port matches the 'Destination' field in your config.

Example: SIMH (VAX/PDP-11)

For a line printer on a VAX (LPT):

```
att lpt 1403
```

(Note: SIMH configuration varies widely depending on the specific machine and line printer emulation).

---

6. Output

Generated files will appear in the **Output Dir** specified for each device.

- **PDFs**: High-resolution, searchable PDFs with authentic greenbar backgrounds, tractor feed holes, and page numbering.
- **Text**: Standard ASCII text files containing the print job content.