TN5250 HOWTO

5250 Emulation For Connecting to IBM iSeries Hosts

James Rich
Salt Lake City, Utah
United States of America

Contents

Section 1
Introduction

1.1 Preface

tn5250 is a 5250 connection package for communication with IBM iSeries hosts. tn5250 includes 5250 display emulation and printing. This HOWTO is written primarily for the GNU/Linux version of tn5250, though other unix-like systems will likely have much in common. The Windows version of tn5250 is less similar, though much useful information can be obtained by the astute reader.

1.2 Copyright information

1.3 Disclaimer

No liability for the contents of this documents can be accepted. Use the concepts, examples and other content at your own risk. As this is a new edition of this document, there may be errors and inaccuracies, that may of course be damaging to your system. Proceed with caution, and although this is highly unlikely, the author(s) do not take any responsibility for that.

All copyrights are held by their by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark.

Naming of particular products or brands should not be seen as endorsements.

You are strongly recommended to take a backup of your system before major installation and backups at regular intervals.

1.4 Document version

This is version 1.1a of the tn5250 HOWTO. The newest version of this HOWTO is available at ftp.chowhouse.com/pub/tn5250. Currently this HOWTO is only available in TEX format, Adobe Portable Document Format (PDF), and Postscript. Other formats will be made available in the future.

1.5 Where to find more information

More information is on the tn5250 website at http://tn5250.sourceforge.net/. There is also a mailing list dedicated to the tn5250 project. In order to send email to the mailing list, you must be subscribed. Subscribing is easy. Just go to http://lists.midrange.com/mailman/listinfo/linux5250 and fill in the required information. The list has fairly low traffic and the people there are friendly and helpful.

1.6 Credits

Thanks to Michael Madore, Jason Felice, Carey Evans, Scott Klement, Rich Duzenbury, Frank Richter, Roger Bowler, Ron Colcernian, Jan Echternach, Colin McCormack, Dave McKenzie, Peter Schlaile, William J. Suetholz, Jean-Marie Guenebaud, Martin Rowe, Daniel Teixeira, and Steven Kurylo for making tn5250 possible and their assistance in creating this document.

Thanks also to David Gibbs and midrange.com, and sourceforge.net for providing resources related to tn5250's development.

1.7 Feedback

Feedback is welcome for this document. Without your submissions and input, this document wouldn't exist. Please send your additions, comments and criticisms to james@eaerich.com.

1.8 Translations

No translations exist at this time. If someone would like to submit a translation please contact me at james@eaerich.com.

Section 2
Obtaining and Installing the Software

2.1 Where to get the software

The latest releases of the software can be obtained from the SourceForge tn5250 project homepage at http://tn5250.sourceforge.net/. There are tarballs available as well as CVS access.

You can download the tarball at

http://prdownloads.sourceforge.net/tn5250/.

To get the latest code you can download the CVS repository by using the following commands:

```
cvs -d :pserver:anonymous@cvs.tn5250.sourceforge.net:/cvsroot/tn5250 login
```

At the password prompt, hit enter

```
cvs -z3 -d :pserver:anonymous@cvs.tn5250.sourceforge.net:/cvsroot/tn5250 checkout tn5250
```

2.2 Configuring the build

After downloading the source there are two ways to build it, depending on whether you downloaded a tarball or checked it out from CVS. If you checked out the code from CVS follow these instructions to build the code:

```
cd tn5250
./autogen.sh
./configure
```

If you downloaded a tarball the steps are almost exactly the same:

```
cd tn5250-<version>      where <version> is the release version
./configure
```

There are two important arguments to the `./configure` script: `--prefix` and `--with-ssl`. By default the software will be installed in `/usr/local`. You can choose a different installation directory when you compile by using `./configure` with the `--prefix` argument. It is important to remember what directory you specify here as key maps will be installed under this directory. If you want SSL support make sure you have OpenSSL installed and pass the `--with-ssl` argument to `./configure`. For example:

```
./configure --prefix=/usr --with-ssl
```

Will configure the build to install in `/usr` with SSL support.

2.3 Compiling and installing the software

Once `./configure` has run compile and install tn5250:

```
make
su
make install
```

Once `make install` is done you are ready to run tn5250.

Section 3
Using tn5250

3.1 Two ways to run tn5250

After installation you have two ways to run tn5250: from the console and within an xterm. If you want to run tn5250 from the console just use:

```
tn5250 as400.mydomain.com
```

where as400.mydomain.com is the fully qualified domain name of your iSeries. For those who run the X Windows System there is also a script called xt5250 which will start tn5250 inside of an xterm. This script is the preferred way to run tn5250 when using X. xt5250 sets a number of required configuration parameters before running tn5250. You can start xt5250 by entering:

```
xt5250 as400.mydomain.com
```

where again as400.mydomain.com is the fully qualified domain name of your iSeries.

3.2 Macros

tn5250 has a macro feature that allows you to record a sequence of keystrokes and play that sequence back at a later time. To begin recording a macro press Control-S followed by the function key you want to use to replay the macro. Then enter the key sequence you want to record. When you are finished, press Control-S again to stop recording. To replay the macro you recorded, press Control-W followed by the function key you selected during the recording process.

The macros you record are stored in a file in your home directory called .tn5250macros. This is a plain text file which means it can be easily edited to change your macros without having to rerecord them. Be aware that you can only put up to 100 characters on a line. Macros are separated by two linefeeds and identified by a function key identifier in square brackets.

Section 4
Configuration

4.1 tn5250rc configuration file

From the previous section we saw that tn5250 can be started giving the fully qualified domain name of the iSeries you want to connect to as an argument at the command prompt. This is not always ideal or the easiest way to do things. Typically users want to connect to the same iSeries every time they use tn5250. It would be nice if tn5250 could remember what host we want to work with and even set some preferences for working with that host.

tn5250 uses two files to remember such settings: a system-wide configuration file and a per user configuration file. The system-wide configuration file is $PREFIX/etc/tn5250rc where $PREFIX is the value of the `./configure` option `--prefix` when the program was compiled (as discussed in subsection 2.2). Configuration options that affect all users should be placed in this file.

Each user can also define their own configuration in the file .tn5250rc in their home directory. It is probably best to put most settings in this file rather than the system-wide configuration file, as most users will probably have their own way of doing things. And if a mistake is made and the .tn5250rc file becomes unusable it can just be deleted without adversely affecting the operation of tn5250.

The syntax of the file is fairly straightforward. Options are set using a `parameter = value` syntax or by a simple boolean `+feature` type syntax:

```
identifier {
    parameter = value
    +feature
    ...
}
```

4.2 Defining host systems

If you want to connect to a host called as400.mydomain.com you would create a minimal .tn5250rc file that looks like this:

```
as400 {
    host = as400.mydomain.com
}
```

Now you can just type:

```
tn5250 as400
```

at the command line.

Any number of hosts may be defined in the .tn5250rc file.

4.3 Display types

The display type that you want to emulate is set using the `env.TERM` parameter. The supported display types are listed in the tn5250 man page. The most common display type to emulate is the IBM-3477-FC. You can specify the display type at the command prompt or add it to your .tn5250rc file. Adding the display type to our sample .tn5250rc file gives the following:

```
as400 {
    host = as400.mydomain.com
    env.TERM = IBM-3477-FC
}
```

4.4 Device names

tn5250 supports device naming. This means that you can specify a device name to use when connecting to the iSeries. Device names are specified using the `env.DEVNAME` argument. Device names may be up to ten characters long. If you want to call your tn5250 session MYDISPLAY then you use:

```
 env.DEVNAME = MYDISPLAY
```

Adding this to our .tn5250rc file results in:

```
as400 {
    host = as400.mydomain.com
    env.TERM = IBM-3477-FC
    env.DEVNAME = MYDISPLAY
}
```

4.5 Console mode underlines

When running in console mode tn5250 by default does not display the underlines that are typically part of most 5250 screens. This is because the console does not provide support for underlines. tn5250 can emulate underlines when started with the `+underscores` argument. This argument should only be given when running tn5250 at the console. When running tn5250 in an xterm (using xt5250) the `+underscores` argument should not be used as xterm provides native support for them.

Adding `+underscores` to our .tn5250rc file looks like this:

```
as400 {
    host = as400.mydomain.com
    env.TERM = IBM-3477-FC
    env.DEVNAME = MYDISPLAY
    +underscores
}
```

4.6 Translation maps

Translation maps define how characters are mapped from EBCDIC to ASCII. You should use the same translation map as your iSeries has defined for interactive jobs. You should check to see what CCSID your iSeries uses. The default for tn5250 is 37. CCSIDs on the iSeries are listed in Appendix G of IBM manual SC41-5101-01, AS/400 National Language Support. This manual is available on the web at http://publib.boulder.ibm.com/cgi-bin/bookmgr/BOOKS/QB3AWC01/G.2.

Currently recognised values for the translation map are: 37, 256, 273, 277, 278, 280, 284, 285, 290, 297, 420, 424, 500, 870, 871, 875, 880, 905, and 1026. Most of these are translated to ISO-8859-1, also known as Latin 1. For more information see the tn5250 man page.

Translation maps are specified using the map argument. To set the map to be 37 use:

```
map = 37
```

With this added to our .tn5250rc file we get:

```
as400 {
    host = as400.mydomain.com
    env.TERM = IBM-3477-FC
    env.DEVNAME = MYDISPLAY
    +underscores
    map = 37
}
```

4.7 Automatic login

You can configure tn5250 to automatically log in to the iSeries using the `env.USER`, `env.IBMSUBSPW`, `env.IBMPROGRAM`, `env.IBMIMENU`, and `env.IBMCURLIB` arguments. If you wanted to automatically log in as user MYUSER with password MYPASSWORD you would set these arguments like this:

```
env.USER = MYUSER
```

and

```
env.IBMSUBSPW = MYPASSWORD
```

The `env.IBMPROGRAM`, `env.IBMIMENU`, and `env.IBMCURLIB` arguments refer to the respective signon fields for program, menu, and current library. These have the same syntax as `env.USER` and `env.IBMSUBSPW`. Adding these to our .tn5250rc file results in:

```
as400 {
    host = as400.mydomain.com
    env.TERM = IBM-3477-FC
    env.DEVNAME = MYDISPLAY
    +underscores
    map = 37
    env.USER = MYUSER
    env.IBMSUBSPW = MYPASSWORD
    env.IBMPROGRAM = MYPROGRAM
    env.IBMMENU = MYMENU
    env.IBMCURLIB = MYLIBRARY
}
```

4.8 Local printing

Usually pressing the print key results in the screen being printed by the iSeries. This behaviour can be changed to have the host on which tn5250 is running do the print screen by using the `+local_print_key` argument. `+local_print_key` will cause a print screen to be sent to the default printer when the print screen key is pressed (usually Control-P).

On unix-like systems there are a number of arguments associated with `+local_print_key` to control how the resulting postscript is generated. These are described in the following table:

Argument	Description	Default
`outputcommand=CMD`	command to pipe postscript to	lpr
`psfontsize_80=NUM`	size in dots of font in 80 columns	10
`psfontsize_132=NUM`	size in dots of font in 132 columns	7
`pagewidth=NUM`	width in dots of page	612
`pagelength=NUM`	length in dots of page	792
`leftmargin=NUM`	left margin of page in dots	18
`topmargin=NUM`	top margin of page in dots	36

Adding `+local_print_key` to our .tn5250rc file looks like this:

```
as400 {
    host = as400.mydomain.com
    env.TERM = IBM-3477-FC
    env.DEVNAME = MYDISPLAY
    +underscores
    map= 37
    env.USER = MYUSER
    env.IBMSUBSPW = MYPASSWORD
    env.IBMPROGRAM = MYPROGRAM
    env.IBMMENU = MYMENU
    env.IBMCURLIB = MYLIBRARY
    +local_print_key
}
```

Section 5
Key Mapping

5.1 Two types of key maps

tn5250 uses one of two types of key maps depending on how it is invoked. The first type is the console key map. When run from the console tn5250 uses a built in keymap. This keymap can be overridden using the `loadkeys` command. Two key maps are included with the tn5250 package for use with the console: us5250.map and uk5250.map. These files can be modified to suit your needs. The `loadkeys` command is a linux-only command. On other platforms you will need to read the platform specific instructions. For FreeBSD these can be found in the README file in the freebsd directory of the tn5250 source distribution. Likewise for Solaris 7 there are instructions in the README.sun file in the sun directory. An example of a Portugese key map for use with the console is included in the appendix.

The second type of key map is used when tn5250 is run using the xt5250 script in an xterm. This key map uses the standard X11 methods for handling key strokes. Because of this it is more flexible and doesn't require using the loadkeys command. The sections that follow cover this second type of key map.

5.2 System-wide key map

The system key map file is stored in $PREFIX/share/tn5250/XTerm where $PREFIX is the value to the `./configure` option `--prefix` when the program was compiled (as described in subsection 2.2). Making changes to this file will affect all the users on your system. This is the correct file to change when you want all users to have the same configuration.

However, it is probably best to not modify this file for several reasons. First, modifications made to this file will affect all users which may not be desireable. Second, if you change the map in a way that makes tn5250 no longer useable you will have to reinstall it. Third, you have to have root access to modify this file which prevents users from making changes. And finally, when you reinstall or upgrade tn5250 it will erase the changes you have made. Fortunately, a better solution exists that solves these problems.

5.3 User key map

A better solution is to put user key maps in your .Xdefaults (or .Xresources) file in your home directory. It doesn't matter which file you use, .Xdefaults or .Xresources, both have the same effect. In fact, you can make one a symbolic link to the other just to avoid confusion.

.Xdefaults and .Xresources are read the startup of your X Window System session and these files change the way xterm behaves. Since they are read only at startup it is necessary to log out and log back in in order for changes to these files to take effect. Because they are read by Xlib and change xterm, this only works when you use xt5250 running in a real xterm. Some people report that aterm also works, but this is not guaranteed. Entries in .Xdefaults or .Xresources will not affect tn5250 when running from the console.

In order to use these files for the key map, we have to add entries for the keys contained in the system-wide key map. For the sake of simplicity we will use .Xdefaults, but these steps apply equally well to .Xresources. First add the contents of the system-wide map by using to following command (remember to substitute the directory specified in the `./configure` option `--prefix` as explained in subsection 2.2. We assume the default of /usr/local):

```
cat /usr/local/share/tn5250/XTerm >> ~/.Xdefaults
```

This will append the default key map into your .Xdefaults file.

5.4 Changing the key map

The key map is written as <key modifier><key>: string("<value>"). The <key modifier> indicates if the Shift key or other modifier key must be pressed. <key> is which key on the keyboard is being pressed. <value> is a value sent to the host. By changing <value> you change what effect a given key (with optional key modifier) has. The tricky part is knowing what string <value> should be.

An example from the XTerm file will help to explain things:

```
~Shift <Key>F1:  string("\0331") \n\
Shift <Key>F1:  string("\033!") \n\
```

The key modifier in the first line is `~Shift`. The ~ symbol means **not**. In other words, the Shift key is *not* pressed. The key is `<Key>F1`. The value given to the program is \0331. The \033 part is the ASCII representation of the Escape key in octal. \0331 means Escape followed by 1. Escape followed by 1 is interpreted on UNIX terminals as the F1 key. You can try this on old terminals or in xterm.

The tn5250 man page lists the strings necessary to generate a 5250 key press. According to the man page the 5250 key Field Exit is Escape followed by X. If you want to make the Enter key on the 10-key keypad map to the 5250 Field Exit key you would put the following in your .Xdefaults file:

```
xt5250*VT100.Translations:  #override\n\
        <Key>KP_Enter:  string("\033X")
```

You can find some useful examples of complete key map files in the appendix.

Section 6
Using Fonts

6.1 Font selection in tn5250rc

You can specify what font to use by placing the font definition in your .tn5250rc file. You can specify fonts for 80 column display and 132 column display. Adding some font settings to our .tn5250rc file results in:

```
as400 {
    host = as400.mydomain.com
    env.TERM = IBM-3477-FC
    env.DEVNAME = MYDISPLAY
    map=37
    env.USER = MYUSER
    env.IBMSUBSPW = MYPASSWORD
    env.IBMPROGRAM = MYPROGRAM
    env.IBMMENU = MYMENU
    env.IBMCURLIB = MYLIBRARY
    +local_print_key
    font_80=-*-fixed-*-*-*-*-*-200-75-100-*-*-*-*
    font_132=-*-fixed-*-*-*-*-*-200-75-100-*-*-*-*
}
```

Note that we have removed the `+underscores` argument since it can't be used with X11.

To determine what fonts you like use the `xlsfonts` or `xfontsel` commands. xlsfonts tells you what fonts you have on your system and xfontsel shows you what the fonts look like.

6.2 Font selection in .Xdefaults

You can also put font settings in your .Xdefaults file. Put a `xt5250*Font :` line in your .Xdefaults file followed by the font you want to use. For example:

```
xt5250*Font :  7x13
```

will give you a fixed 7x13 font.

Section 7
Changing Screen Colors

Currently the only colors you can change without making source code modifications are the color of the cursor and pointer in xt5250. Changeable colors is on the TODO list.

You can set the cursor and pointer colors in your .Xdefaults by specifying:

```
xt5250*cursorColor :  gold
xt5250*pointerColor :  yellow
```

In this example we set the 5250 cursor to a gold color and the mouse pointer to yellow.

Section 8
Using tn5250 With SSL

8.1 Configuring the Digital Certificate Manager

Before you can use SSL with tn5250 you must configure the iSeries to enable SSL support. This is done using the Digital Certificate Manager (DCM, part of the Cryptographic Access Provider). The DCM is not supported on all releases. The DCM first became available on V4R5. The information provided here is accurate up to version V5R2 of OS/400. You should consult the IBM provided documentation for your release.

In order to use the DCM you must have at a minimum the following OS/400 products installed on your iSeries:

5722SS1 option 34 — OS/400 – Digital Certificate Manager

5722SS1 option 35 — OS/400 – Cryptographic Service Provider

5722AC3 — Crypto Access Provider 128-bit for AS/400

5722DG1 — IBM HTTP Server

5722JV1 option 3 — Java Developer Kit 1.2

There are several steps required to create a digital certificate. Most of these steps need only to be done once. The Digital Certificate Manager is accessed by pointing a web browser to your iSeries hostname and port 2001 like this: http://myas400.mydomain.com:2001/. This will access the *ADMIN web server on the iSeries. See the IBM documention for information on configuring the DCM.

Once the digital certificate has been created you need to download the certificate to your computer. The DCM provides two ways to this: install in a browser and copy and paste. We want to use the copy and paste method. Following the copy and paste link results in a page being displayed that shows the digital certificate. Copy everything after and including the `-----BEGIN CERTIFICATE-----` line up to and including the `-----END CERTIFICATE-----` line. Save this to a file and give it some meaningful name, such as `myas400.cert`.

8.2 Enabling SSL support in tn5250

You must compile tn5250 with SSL support to use SSL. See subsection 2.2 *Configuring the build* for information on building tn5250 with SSL support.

Once compiled with SSL support, tn5250 is configured quite easily. Use `+ssl_verify_server` in your .tn5250rc file to turn on SSL authentication. You must also prepend `ssl:` to the `host` parameter in your .tn5250rc file.

tn5250 also uses the certificate generated by the iSeries Digital Certificate Manager for use when authenticating SSL sessions. The certificate may be stored anywhere on your system and is the certificate saved in the previous subsection. This certificate is referenced using the `ssl_ca_file` argument. If you have configured your iSeries to only accept SSL sessions from authenticated clients you will need this file.

Adding `+ssl_verify_server` and `ssl_ca_file` and prepending `ssl:` to the `host` in our .tn5250rc file results in:

```
as400 {
    host = ssl:as400.mydomain.com
    env.TERM = IBM-3477-FC
    env.DEVNAME = MYDISPLAY
    +underscores
    map=37
    env.USER = MYUSER
    env.IBMSUBSPW = MYPASSWORD
    env.IBMPROGRAM = MYPROGRAM
    env.IBMMENU = MYMENU
    env.IBMCURLIB = MYLIBRARY
    +local_print_key
    font_80=-*-fixed-*-*-*-*-*-200-75-100-*-*-*-*
    font_132=-*-fixed-*-*-*-*-*-200-75-100-*-*-*-*
    +ssl_verify_server
    +ssl_ca_file = /path/to/certificate/file
}
```

In the example above, `/path/to/certificate/file` refers to the location where you stored the file that contains the certificate generated by the Digital Certificate Manager.

Section 9
Printing With lp5250d

9.1 Basic structure

Printing iSeries output with lp5250d involves two basic steps: starting a printer daemon that connects to the iSeries and indicates it is ready for print jobs and converting the output received from the iSeries into a printable format.

The first step is done by the lp5250d daemon. When lp5250d starts it connects to the specified iSeries and starts a writer with the name passed in the `env.DEVNAME` argument. lp5250d receives the printed output from the writer, but cannot print it directly. The data as sent from the iSeries is EBCDIC encoded and formatted for an iSeries printer (this is the *SCS parameter in the print file description). The data must be converted to ASCII and reformatted to something the destination printer can use.

The conversion to ASCII and reformatting is the second step in printing iSeries data. This is usually done by one of three programs: scs2ascii, scs2pdf, and scs2ps. These programs are passed as a parameter to lp5250d as part of the `outputcommand` argument. lp5250d pipes the output sent to it from the writer to whatever command is passed in the `outputcommand` argument. It is also possible to directly output the EBCDIC print stream to a file. scs2ascii translates the EBCDIC encoded *SCS stream into plain ASCII text. scs2pdf does the same but outputs a PDF document. Likewise scs2ps outputs a postscript file.

lp5250d can only print *SCS data at this time. This isn't due to lp5250d itself, but rather to the fact that there isn't at this time a translator for other output types. If such a translator existed lp5250d could use it by passing it on the `outputcommand` parameter.

9.2 Using lp5250d

lp5250d is started from the command line like this:

```
lp5250d env.DEVNAME=MYPRINT outputcommand="scs2ascii | lpr" as400.mydomain.com
```

In this example `MYPRINT` is the name of the printer that you want to use on the iSeries. The `outputcommand` argument is passed a complete command, in this case `"scs2ascii | lpr"`. Once again as400.mydomain.com is the fully qualified domain name of your iSeries.

It is important that the parameter passed in the `outputcommand` argument be a complete command as lp5250d simply invokes whatever is passed here and pipes the iSeries output to it. As we saw in the preceding section this enables you to do whatever you want with the printed output sent from the iSeries. For example, if you want to simply send the printed output directly to a file without converting it to ASCII you could change the `outputcommand` argument to something like this:

```
outputcommand="cat > /tmp/output.scs"
```

The resulting file `/tmp/output.scs` would contain the data exactly as lp5250d received it from the iSeries. You could at this point print it out with a command like:

```
cat /tmp/output.scs | scs2asii | lpr
```

If you want to save the output to a file use `scs2pdf` or `scs2ps` with `outputcommand`. These commands format the printed output into Adobe Portable Document Format (PDF) and postscript respectively. scs2pdf creates searchable PDFs with the name given in the `outputcommand` argument. scs2ps can also be directed to a file, but is more commonly used to print to postscript printers.

You can create dynamic filenames by placing '$$' in the filename given to `outputcommand`. The '$$' will be replaced with the process ID of the command passed to `outputcommand` when the print job is run.

This doesn't strictly gaurantee that the filename will always be unique as process IDs can wrap. To add the process ID to the filename use `outputcommand` like this:

```
outputcommand="scs2pdf > /tmp/output$$.pdf"
```

Another idea is to add the current date to the filename. This is done easily using the unix `date` command like so:

```
outputcommand="scs2pdf > /tmp/output-‘ date +%m-%d-%H-%M-%S‘.pdf"
```

9.3 lp5250d and .tn5250rc

You can save your lp5250d settings in $HOME/.tn5250rc just like you can with tn5250. To do so, create a new identifier to hold the settings. The new identifier can reference the same iSeries, but you will only use it with lp5250d. The syntax is the same as we used with tn5250, so our example is straightforward:

```
asciiprint {
    host = as400.mydomain.com
    env.DEVNAME = MYPRINT
    outputcommand = scs2ascii | lpr
}
```

Then simply invoke lp5250d like this:

```
lp5250d asciiprint
```

9.4 Specifying a printer message queue

You can specify a message queue to receive printer related messages using the `env.IBMMSGQNAME` and `env.IBMMSGQLIB` arguments. This way you can direct messages such as form type changes to a given workstation. For example, if you want to send printer messages for printer MYPRINT to display DSP01 you would make your .tn5250rc file look like this:

```
asciiprint {
    host = as400.mydomain.com
    env.DEVNAME = MYPRINT
    outputcommand = scs2ascii | lpr
    env.IBMMSGQNAME = DSP01
    env.IBMMSGQLIB = *LIBL
}
```

9.5 Using Host Print Transform

Host Print Transform causes the iSeries to format the printed output to a specified printer before sending it to lp5250d. This is useful for printing complex spooled files like *AFPDS and *IPDS print files on printers that don't support those languages. The drawback is that Host Print Transform can only be used to actually print and not for writing to a file. Since the print stream is received as a printer language it is really only useful when sent directly to a printer. The formatting programs scs2pdf and scs2ps don't understand printer language and therefore can't be used to create those types of documents from complex printer ouput.

The list of printers that Host Print Transform can use is found by entering the OS/400 command CRTDEVPRT and pressing F4. Look for the list of available “manufacturer types and models” in the MFRTYPMDL parameter. A complete list of available printers for an iSeries running V5R1 is listed in the appendix.

Host Print Transform is usually used in conjuction with scs2ascii which simply escapes the printer control language and translates only the text. The result is then piped to lpr. Host Print Transform is specified using the `env.IBMMFRTYPMDL` argument. Changing the .tn5250rc file to use Print Transform to output to a Hewlett-Packard LaserJet 4 printer is simple:

```
asciiprint {
    host = as400.mydomain.com
    env.DEVNAME = MYPRINT
    outputcommand = scs2ascii | lpr
    env.IBMMSGQNAME = DSP01
    env.IBMMSGQLIB = *LIBL
    env.IBMMFRTYPMDL = *HP4
}
```

9.6 Defining printer languages using workstation customization

Workstation customization (WSCST) is a method to define a printer language using the Host Print Transform mechanism. If none of the available printers listed on the CRTDEVPRT command as described in the previous subsection meets your needs then you can define a custom object to use with the `env.IBMMFRTYPMDL` argument. WSCST objects are specified using the `env.IBMMFRTYPMDL` argument, just as Host Print Transform, and `env.IBMWSCSTNAME` and `env.IBMWSCSTLIB`. To use workstation customization with the standard IBM *SCS to ASCII translator you would use:

```
    env.IBMMFRTYPMDL = *WSCST
    env.IBMWSCSTNAME = QWPDEFAULT
    env.IBMWSCSTLIB = *LIBL
```

This has the same effect as using scs2ascii without any other arguments.

To create your own printer language definition you can copy the IBM supplied definitions for *HP4 and then change those. Use the following OS/400 command to retrieve the *HP4 definitions into a member called HP4 in source file QTXTSRC in library MYLIB:

```
 RTVWSCST DEVTYPE(*TRANSFORM) MFRTYPMDL(*HP4) SRCFILE(MYLIB/QTXTSRC) SRCMBR(HP4)
```

Now edit the HP4 member of MYLIB/QTXTSRC to suit your needs. Once you have finished making your changes create the WSCST object with this OS/400 command:

```
 CRTWSCST WSCST(MYLIB/MYWSCST) SRCFILE(MYLIB/QTXTRC) SRCMBR(HP4)
```

Once that is complete you can reference the new object like this:

```
    env.IBMMFRTYPMDL = *WSCST
    env.IBMWSCSTNAME = MYWSCST
    env.IBMWSCSTLIB = MYLIB
```

Adding this to .tn5250rc results in:

```
asciiprint {
    host = as400.mydomain.com
    env.DEVNAME = MYPRINT
    outputcommand = scs2ascii | lpr
    env.IBMMSGQNAME = DSP01
    env.IBMMSGQLIB = *LIBL
    env.IBMMFRTYPMDL = *WSCST
    env.IBMWSCSTNAME = MYWSCST
    env.IBMWSCSTLIB = MYLIB
}
```

Section 10
Appendix

10.1 Portuguese terminal key map

This is a (nearly) complete keymap for Portugese. Some characters are missing due to current inabilities to print them. These ommisions are indicated by the words “not complete”.

```
# pt.map
# Baseado no portugues.map de lacyp@unicorn.it.wsu.edu 18-jun-1996
# (tomado da distribuicao Mini-Linux). Correcoes de aeb (Andries Brouwer ?).
#
# Adicionado suporte para acentuação por Carlos A M dos Santos,
# <casantos@cpmet.ufpel.tche.br>
#
# 26-dez-1997
# Adicionados comentários e correções.
#
# Este arquivo é distribuído sob os termos da GNU GPL versão 2.
#
charset "iso-8859-1"
keymaps 0-15

keycode    0 =
keycode    1 = Escape         Escape
        alt       keycode       1 = Meta_Escape
        shift     keycode       1 = Control_q
keycode    2 = one            exclam          onesuperior exclamdown
        alt       keycode       2 = Meta_one
        alt shift keycode       2 = Meta_exclam
keycode    3 = two            quotedbl        at twosuperior
        control   keycode       3 = nul
        control shift keycode   3 = nul
        alt       keycode       3 = Meta_two
        alt shift keycode       3 = Meta_quotedbl
        alt altgr keycode       3 = Meta_at
keycode    4 = three          numbersign      pound threesuperior
        control   keycode       4 = Escape
        alt       keycode       4 = Meta_three
        alt shift keycode       4 = Meta_numbersign
keycode    5 = four           dollar          section
        control   keycode       5 = Control_backslash
        alt       keycode       5 = Meta_four
        alt shift keycode       5 = Meta_dollar
keycode    6 = five           percent         cent
        control   keycode       6 = Control_bracketright
        alt       keycode       6 = Meta_five
        alt shift keycode       6 = Meta_percent
keycode    7 = six            ampersand
        alt       keycode       7 = Meta_six
        alt shift keycode       7 = Meta_ampersand
```

```
keycode    8 = seven          slash          braceleft
         alt      keycode      8 = Meta_seven
         alt shift keycode     8 = Meta_slash
         alt altgr keycode     8 = Meta_braceleft
keycode    9 = eight          parenleft      bracketleft
         control  keycode      9 = Delete
         control altgr keycode 9 = Escape
         alt      keycode      9 = Meta_eight
         alt shift keycode     9 = Meta_parenleft
         alt altgr keycode     9 = Meta_bracketleft
keycode   10 = nine           parenright     bracketright
         alt      keycode     10 = Meta_nine
         alt shift keycode    10 = Meta_parenright
         alt altgr keycode    10 = Meta_bracketright
keycode   11 = zero           equal          braceright
         alt      keycode     11 = Meta_zero
         alt shift keycode    11 = Meta_equal
         alt altgr keycode    11 = Meta_braceright
keycode   12 = apostrophe     question       backslash
         alt      keycode     12 = Meta_apostrophe
         alt shift keycode    12 = Meta_question
keycode   13 = guillemotleft  guillemotright
keycode   14 = Delete         Delete         Delete
         alt      keycode     14 = Meta_Delete
keycode   15 = Tab            Tab
         alt      keycode     15 = Meta_Tab
keycode   16 = q
keycode   17 = w
keycode   18 = e
keycode   19 = r
         altgr    keycode     19 = registered
keycode   20 = t
keycode   21 = y
keycode   22 = u
keycode   23 = i
keycode   24 = o
keycode   25 = p
keycode   26 = plus           asterisk       dead_diaeresis
         alt      keycode     26 = Meta_plus
         alt shift keycode    26 = Meta_asterisk
keycode   27 = dead_acute     dead_grave     acute         grave
         control  keycode     27 = nul
         alt      keycode     27 = Meta_apostrophe
         alt shift keycode    27 = Meta_grave
keycode   28 = Control_k
         alt      keycode     28 = Meta_Control_m
keycode   29 = Control_r
keycode   30 = a
keycode   31 = s
keycode   32 = d
keycode   33 = f
keycode   34 = g
keycode   35 = h
keycode   36 = j
keycode   37 = k
keycode   38 = l
keycode   39 = +ccedilla       +Ccedilla
keycode   40 = masculine       ordfeminine
keycode   41 = backslash       bar
         control   keycode     41 = Control_backslash
         alt       keycode     41 = Meta_backslash
         alt shift keycode     41 = Meta_bar
keycode   42 = Shift
```

    #keycode 42 = Control_q

```
keycode   43 = dead_tilde      dead_circumflex  asciitilde        asciicircum
         control   keycode     43 = nul
         control shift keycode 43 = Control_asciicircum
         alt       keycode     43 = Meta_asciicircum
keycode   44 = z
keycode   45 = x
keycode   46 = c
keycode   47 = v
keycode   48 = b
keycode   49 = n
keycode   50 = m
keycode   51 = comma           semicolon
         alt       keycode     51 = Meta_comma
keycode   52 = period          colon
         alt       keycode     52 = Meta_period
keycode   53 = minus           underscore
         control shift keycode 53 = Control_underscore
         control   keycode     53 = Delete
         alt       keycode     53 = Meta_minus
         alt shift keycode     53 = Meta_underscore
keycode   54 = Shift
keycode   55 = KP_Multiply
keycode   56 = Alt
keycode   57 = space           space
         control   keycode     57 = nul
         alt       keycode     57 = Meta_space
keycode   58 = Caps_Lock
keycode   59 = F1              F13              Console_13
         control   keycode     59 = F1
         alt       keycode     59 = Console_1
         control alt keycode   59 = Console_1
keycode   60 = F2              F14              Console_14
         control   keycode     60 = F2
         alt       keycode     60 = Console_2
         control alt keycode   60 = Console_2
keycode   61 = F3              F15              Console_15
         control   keycode     61 = F3
         alt       keycode     61 = Console_3
         control alt keycode   61 = Console_3
keycode   62 = F4              F16              Console_16
         control   keycode     62 = F4
         alt      keycode     62 = Console_4
         control alt keycode   62 = Console_4
keycode   63 = F5             F17             Console_17
         control  keycode     63 = F5
         alt      keycode     63 = Console_5
         control alt keycode   63 = Console_5
keycode   64 = F6             F18             Console_18
         control  keycode     64 = F6
         alt      keycode     64 = Console_6
         control alt keycode   64 = Console_6
keycode   65 = F7             F19             Console_19
         control  keycode     65 = F7
         alt      keycode     65 = Console_7
         control alt keycode   65 = Console_7
keycode   66 = F8             F20             Console_20
         control  keycode     66 = F8
         alt      keycode     66 = Console_8
         control alt keycode   66 = Console_8
keycode   67 = F9             F21             Console_21
         control  keycode     67 = F9
         alt      keycode     67 = Console_9
         control alt keycode   67 = Console_9
keycode   68 = F10            F22             Console_22
         control  keycode     68 = F10
         alt      keycode     68 = Console_10
         control alt keycode   68 = Console_10
keycode   69 = Num_Lock
keycode   70 = Scroll_Lock    Show_Memory     Show_Registers
         control  keycode     70 = Show_State
         alt      keycode     70 = Scroll_Lock
keycode   71 = KP_7
         alt      keycode     71 = Ascii_7
keycode   72 = KP_8
         alt      keycode     72 = Ascii_8
keycode   73 = KP_9
         alt      keycode     73 = Ascii_9
keycode   74 = KP_Subtract
keycode   75 = KP_4
         alt      keycode     75 = Ascii_4
keycode   76 = KP_5
         alt      keycode     76 = Ascii_5
keycode   77 = KP_6
         alt      keycode     77 = Ascii_6
keycode   78 = KP_Add
keycode   79 = KP_1
         alt      keycode     79 = Ascii_1
keycode   80 = KP_2
         alt      keycode     80 = Ascii_2
keycode   81 = KP_3
         alt      keycode     81 = Ascii_3
keycode   82 = KP_0
         alt      keycode     82 = Ascii_0
keycode   83 = KP_Period
```

```
         altgr control keycode  83 = Boot
         control alt keycode    83 = Boot
keycode   84 = Last_Console
keycode   85 =
keycode   86 = less           greater          backslash
         alt      keycode      86 = Meta_less
         alt shift keycode     86 = Meta_greater
keycode   87 = F11            F23              Console_23
         control  keycode      87 = F11
         alt      keycode      87 = Console_23
         control alt keycode   87 = Console_11
keycode   88 = F12            F24              Console_24
         control  keycode      88 = F12
         alt      keycode      88 = Console_24
         control alt keycode   88 = Console_12
keycode   89 =
keycode   90 =
keycode   91 =
keycode   92 =
keycode   93 =
keycode   94 =
keycode   95 =
keycode   96 = KP_Enter
keycode   97 = Return
#
# Comente a linha anterior e remova o comentário da linha a seguir para
# atribuir a função Compose à tecla Control direita.
#
#keycode 97 = Compose

keycode   98 = KP_Divide
keycode   99 = Control_backslash
         control  keycode      99 = Control_backslash
         alt      keycode      99 = Control_backslash
keycode  100 = AltGr
keycode  101 = Break
#
# Find é um sinônimo para Home no console, o que não ocorre no X.
#

keycode  102 = Find
keycode  103 = Up
keycode  104 = Prior
         shift    keycode     104 = Scroll_Backward
keycode  105 = Left
         alt      keycode     105 = Decr_Console
keycode  106 = Right
         alt      keycode     106 = Incr_Console
#
# Select é um sinônimo para End no console, o que não ocorre no X.
#
```

```
keycode  107 = Select
keycode  108 = Down
keycode  109 = Next
         shift     keycode     109 = Scroll_Forward
keycode  110 = Insert
keycode  111 = Remove
         altgr control keycode 111 = Boot
         control alt keycode   111 = Boot
keycode  112 =
keycode  113 =
keycode  114 =
keycode  115 =
keycode  116 =
keycode  117 =
keycode  118 =
keycode  119 =
keycode  120 =
keycode  121 =
keycode  122 =
keycode  123 =
keycode  124 =
keycode  125 =

#
# Para os teclados padrão "Windows 95" podemos atribuir a função Compose
# à tecla "janela direita":

keycode  126 = Compose
keycode  127 =

string F1 = "\033[[A"
string F2 = "\033[[B"
string F3 = "\033[[C"
string F4 = "\033[[D"
string F5 = "\033[[E"
string F6 = "\033[17~"
string F7 = "\033[18~"
string F8 = "\033[19~"
string F9 = "\033[20~"
string F10 = "\033[21~"
string F11 = "\033[23~"
string F12 = "\033[24~"
string F13 = "\033[25~"
string F14 = "\033[26~"
string F15 = "\033[28~"
string F16 = "\033[29~"
string F17 = "\033[31~"
string F18 = "\033[32~"
string F19 = "\033[33~"
string F20 = "\033[34~"
string Find = "\033[1~"
string Insert = "\033[2~"
string Remove = "\033[3~"
string Select = "\033[4~"
string Prior = "\033[5˜"
string Next = "\033[6˜"
#alterado por daniel
string F21 = "\033[35˜"
string F22 = "\033[36˜"
string F23 = "\033[37˜"
string F24 = "\033[38˜"
string F25 = ""
string F26 = ""
#
# Se quisermos o mesmo comportamento do X (ou de um terminal VT-*,
# para ser mais exato) podemos usar as definições seguintes.
# Cuidado! Isso pode confundir algumas aplicações.
#
# Observe que no console Home é apenas um sinônimo para Find, assim
# como End é para Select.
# string Home = "\033[7˜"
# string End = "\033[8˜"

# TODAS a definições a seguir são necessárias, pois tendo sido definida
# uma regra de composição, todas as outras são perdidas!
compose ’‘’ ’A’ to ’À’
compose ’‘’ ’a’ to ’à’
compose ’\’’ ’A’ to ’Á’
compose ’\’’ ’a’ to ’á’
compose ’ˆ’ ’A’ to ’Â’
compose ’ˆ’ ’a’ to ’â’
compose ’˜’ ’A’ to ’Ã’
compose ’˜’ ’a’ to ’ã’
compose ’"’ ’A’ to ’Ä’
compose ’"’ ’a’ to ’ä’
compose ’O’ ’A’ to ’Å’
compose ’o’ ’a’ to ’å’
compose ’0’ ’A’ to ’Å’
compose ’0’ ’a’ to ’å’
compose ’A’ ’A’ to ’Å’
compose ’a’ ’a’ to ’å’
compose ’A’ ’E’ to ’Æ’
compose ’a’ ’e’ to ’æ’
compose ’,’ ’C’ to ’Ç’
compose ’,’ ’c’ to ’ç’
compose ’’’ ’C’ to ’Ç’
compose ’’’ ’c’ to ’ç’
compose ’‘’ ’E’ to ’È’
compose ’‘’ ’e’ to ’è’
compose ’\’’ ’E’ to ’É’
compose ’\’’ ’e’ to ’é’
compose ’ˆ’ ’E’ to ’Ê’
compose ’ˆ’ ’e’ to ’ê’
compose ’"’ ’E’ to ’Ë’
compose ’"’ ’e’ to ’ë’
compose ’‘’ ’I’ to ’Ì’
compose ’‘’ ’i’ to ’ì’
compose ’\’’ ’I’ to ’Í’
compose ’\’’ ’i’ to ’í’
compose ’ˆ’ ’I’ to ’Î’
compose ’ˆ’ ’i’ to ’î’
compose ’"’ ’I’ to ’Ï’
compose ’"’ ’i’ to ’ï’
compose ’-’ ’D’ to ’’not complete
compose ’-’ ’d’ to ’’not complete
compose ’˜’ ’N’ to ’Ñ’
compose ’˜’ ’n’ to ’ñ’
compose ’‘’ ’O’ to ’Ò’
compose ’‘’ ’o’ to ’ò’
compose ’\’’ ’O’ to ’Ó’
compose ’\’’ ’o’ to ’ó’
compose ’ˆ’ ’O’ to ’Ô’
compose ’ˆ’ ’o’ to ’ô’
compose ’˜’ ’O’ to ’Õ’
compose ’˜’ ’o’ to ’õ’
compose ’"’ ’O’ to ’Ö’
compose ’"’ ’o’ to ’ö’
compose ’/’ ’O’ to ’Ø’
compose ’/’ ’o’ to ’ø’
compose ’‘’ ’U’ to ’Ù’
compose ’‘’ ’u’ to ’ù’
compose ’\’’ ’U’ to ’Ú’
compose ’\’’ ’u’ to ’ú’
compose ’ˆ’ ’U’ to ’Û’
compose ’ˆ’ ’u’ to ’û’
compose ’"’ ’U’ to ’Ü’
compose ’"’ ’u’ to ’ü’
compose ’\’’ ’Y’ to ’Ý’
compose ’\’’ ’y’ to ’ý’
compose ’T’ ’H’ to ’’not complete
compose ’t’ ’h’ to ’’not complete
compose ’s’ ’s’ to ’ß’
compose ’"’ ’y’ to ’ÿ’
compose ’s’ ’z’ to ’ß’
compose ’i’ ’j’ to ’ÿ’
```

10.2 Key map using XTerm

This is a sample key map using XTerm. This example can either replace the default keymap installed by tn5250 or be appended to your .Xdefaults (or .Xresources) file in your home directory.

```
xt5250*VT100.Translations: #override\n\
        <Key>KP_Add: string("\030") \n\
        <Key>KP_Subtract: string("\033M") \n\
        <Key>KP_Multiply: string("*") \n\
        <Key>KP_Divide: string("/") \n\
        <Key>KP_Enter: string("\033OM") \n\
        <Key>BackSpace: string("\177") \n\
        <Key>End: string("\033[4˜") \n\
        <Key>KP_0: string("0") \n\
        <Key>KP_1: string("1") \n\
        <Key>KP_2: string("2") \n\
        <Key>KP_3: string("3") \n\
        <Key>KP_4: string("4") \n\
        <Key>KP_5: string("5") \n\
        <Key>KP_6: string("6") \n\
        <Key>KP_7: string("7") \n\
        <Key>KP_8: string("8") \n\
        <Key>KP_9: string("9") \n\
        <Key>Scroll_Lock: string("\033H") \n\
        ~Shift<Key>Print: string("\020") \n\
        <Key>Control_L: string("\022") \n\
        <Key>Escape: string("\001") \n\
        Shift <Key>Print: string("\003") \n\
        ~Shift <Key>F1: string("\0331") \n\
        ~Shift <Key>F2: string("\0332") \n\
        ~Shift <Key>F3: string("\0333") \n\
        ~Shift <Key>F4: string("\0334") \n\
        ~Shift <Key>F5: string("\0335") \n\
        ~Shift <Key>F6: string("\0336") \n\
        ~Shift <Key>F7: string("\0337") \n\
        ~Shift <Key>F8: string("\0338") \n\
        ~Shift <Key>F9: string("\0339") \n\
        ~Shift <Key>F10: string("\0330") \n\
        ~Shift <Key>F11: string("\033-") \n\
        ~Shift <Key>F12: string("\033=") \n\
        Shift <Key>F1: string("\033!") \n\
        Shift <Key>F2: string("\033@") \n\
        Shift <Key>F3: string("\033#") \n\
        Shift <Key>F4: string("\033$") \n\
        Shift <Key>F5: string("\033%") \n\
        Shift <Key>F6: string("\033^") \n\
        Shift <Key>F7: string("\033&") \n\
        Shift <Key>F8: string("\033*") \n\
        Shift <Key>F9: string("\033(") \n\
        Shift <Key>F10: string("\033)") \n\
        Shift <Key>F11: string("\033_") \n\
        Shift <Key>F12: string("\033+") \n\
        Shift <Key>Tab: string("\033[Z") \n\
        Shift <Key>Left: string("\033\010") \n\
        Shift <Key>Right: string("\033\025") \n\
        Shift <Key>Home: string("\033\006") \n\
        <Key>Home: string("\033[1~") \n\
        ~Shift <Key>Delete: string("\033[3~") \n\
        ~Shift <Key>Insert: string("\033[2~")
```

10.3 Complete .Xdefaults file with custom colors, fonts, and key map

This is the complete .Xdefaults file I use with xt5250. I made the cursor gold and the mouse pointer yellow. I selected a fixed 7x13 font. I also forced xterm to leave off the left scroll bar and indicated that this is not a login shell.

```
xt5250*cursorColor : gold
xt5250*pointerColor : yellow
```

```
xt5250*Font : 7x13
xt5250*loginShell : off
xt5250*scrollBar : off
xt5250*VT100.Translations: #override\n\
        <Key>KP_Enter: string("\033OM") \n\
        <Key>BackSpace: string("\177") \n\
        <Key>End: string("\033[4~") \n\
        <Key>Scroll_Lock: string("\033H") \n\
        ~Shift<Key>Print: string("\020") \n\
        <Key>Escape: string("\001") \n\
        Shift <Key>Print: string("\003") \n\
        ~Shift <Key>F1: string("\033[[A") \n\
        ~Shift <Key>F2: string("\033[[B") \n\
        ~Shift <Key>F3: string("\033[[C") \n\
        ~Shift <Key>F4: string("\033[[D") \n\
        ~Shift <Key>F5: string("\033[[E") \n\
        ~Shift <Key>F6: string("\033[17~") \n\
        ~Shift <Key>F7: string("\033[18~") \n\
        ~Shift <Key>F8: string("\033[19~") \n\
        ~Shift <Key>F9: string("\033[20~") \n\
        ~Shift <Key>F10: string("\033[21~") \n\
        ~Shift <Key>F11: string("\033[23~") \n\
        ~Shift <Key>F12: string("\033[24~") \n\
        Shift <Key>F1: string("\033[25~") \n\
        Shift <Key>F2: string("\033[26~") \n\
        Shift <Key>F3: string("\033[28~") \n\
        Shift <Key>F4: string("\033[29~") \n\
        Shift <Key>F5: string("\033[31~") \n\
        Shift <Key>F6: string("\033[32~") \n\
        Shift <Key>F7: string("\033[33~") \n\
        Shift <Key>F8: string("\033[34~") \n\
        Shift <Key>F9: string("\033[35~") \n\
        Shift <Key>F10: string("\033[36~") \n\
        Shift <Key>F11: string("\033[37~") \n\
        Shift <Key>F12: string("\033[38~") \n\
        Shift <Key>Tab: string("\033[Z") \n\
        Shift <Key>Left: string("\033\010") \n\
        Shift <Key>Right: string("\033\025") \n\
        Shift <Key>Home: string("\033\006") \n\
        <Key>Home: string("\033[1~") \n\
        ~Shift <Key>Delete: string("\033[3~") \n\
        ~Shift <Key>Insert: string("\033[2~")
```

10.4 Sample tn5250rc

This is a portion of the .tn5250rc file I use. Only the hostnames, usernames, and passwords have been changed.

```
host1 {
    host = host1.mydomain.com
    env.TERM = IBM-3477-FC
    env.USER = MYUSER
    env.IBMSUBSPW = MYPASSWORD
    +local_print_key
}

host2 {
    host = ssl:host2.mydomain.com
    env.TERM = IBM-3477-FC
    +local_print_key
    +ssl_verify_server
    ssl_ca_file = /home/user/host2/host2.cert
}

host2-print {
    host = ssl:host2.mydomain.com
    env.DEVNAME = PRTASCII
    env.IBMMFRTYPMDL = *IBM3116
    +ssl_verify_server
    ssl_ca_file = /home/user/host2/host2.cert
    outputcommand = scs2ascii | lpr
}
```

10.5 Possible values for use with Host Print Transform on V5R1

Below are the possible values that can be used with Host Print Transform on a V5R1 iSeries.

*IBM2380
*IBM2381
*IBM2390
*IBM2391
*IBM3112
*IBM3116
*IBM3130
*IBM3812
*IBM3816
*IBM3912HP
*IBM3916HP
*IBM39302
*IBM39303
*IBM42021
*IBM42022
*IBM42023
*IBM42071
*IBM42072
*IBM42081
*IBM42082
*IBM4212
*IBM4216
*IBM4226
*IBM4230
*IBM4232
*IBM4244MAN
*IBM4019
*IBM4019HP
*IBM4029
*IBM4029HP
*IBM4037
*IBM4039HP
*IBM4070
*IBM4070EP
*IBM4072
*IBM4076
*IBM42011
*IBM42012
*IBM42013
*IBM4244ASF
*IBM4244DUAL
*IBM4247MAN
*IBM4247ASF
*IBM4247DUAL
*IBM4308
*IBM4312
*IBM4317
*IBM4320
*IBM4322
*IBM4324
*IBM4332
*IBM4340

*IBM47121
*IBM47122
*IBM47221
*IBM47222
*IBM4770
*IBM4912
*IBM5152
*IBM5201
*IBM5202
*IBM5204
*IBM5216
*IBM5575
*IBMPAGES
*INFOPRINT21
*INFOPRINT32
*INFOPRINT40
*INFOPRINT2000
*INFOPRINT8C
*CPQPM15
*CPQPM20
*HPII
*HPIID
*HPIIP
*HPIII
*HPIIID
*HPIIIP
*HP310
*HP320
*HP500
*HP520
*HP540
*HP550C
*HP560C
*HPPAINT
*LEXOPTRA
*LEXOPTRAC
*LEXOPTRAN
*LEXOPTRAS
*LEXOPTRASC
*EPFX850
*EPFX870
*EPFX1170
*EPLQ510
*EPLQ570
*EPLQ860
*EPLQ870
*EPLQ1070
*EPLQ1170
*EPLQ2550
*EPLX810
*EPSQ870
*EPSQ1170
*NONE
*IBMPAGESNPB
*IBMPAGES300
*IBMPAGES300NPB
*IBM6400
*IBM6400EP
*IBM6404
*IBM6404EP
*IBM6408
*IBM6408EP
*IBM6412
*IBM6412EP
*INFOPRINT12
*INFOPRINT20
*HPIIISI
*HP4
*HP5
*HP5SI
*HP6
*HP1100
*HP4000
*HP5000
*HP8000
*HPDBCS
*HPCOLORLJ
*HP1200C
*HP1600C
*LEXOPTRAT
*LEXOPTRAW
*LEX2380
*LEX2381
*LEX2390
*LEX2391
*LEX4227
*EPAP2250
*EPAP3250
*EPAP5000
*EPAP5500
*EPDFX5000
*EPDFX8000
*EPEPL7000
*EPEPL8000
*ESCPDBCS
*NECP2
*NECP2200
*NECP2200XE
*NECP5200
*NECP5300
*NECP6200
*NECP6300
*NECPCPR201
*CANLIPS3
*CANLIPS3NPB
*OKI3410

*OKI184IBM
*OKI320IBM
*OKI321IBM
*OKI390IBM
*OKI391IBM
*OKI393IBM
*OKI590IBM
*OKI591IBM
*OKI400
*OKI800
*OKI810
*OKI820
*PAN2624EP
*PAN4410HP
*PAN4420HP
*PAN4430HP
*PAN4450IHP
*PAN4451HP
*XRX4215MRP
*XRX4219MRP
*XRX4220MRP
*XRX4230MRP
*XRX4235
*XRX4700II
*WSCSTLETTER
*PAN1123EP
*PAN1124EP
*PAN1124IEP
*PAN1180EP
*PAN1180IEP
*PAN1191EP
*PAN1624EP
*PAN1654EP
*PAN1695EP
*PAN2123EP
*PAN2124EP
*PAN2180EP
*WSCSTLEGAL
*WSCSTEXECUTIVE
*WSCSTLEDGER
*WSCSTA3
*WSCSTA4
*WSCSTA5
*WSCSTB4
*WSCSTB5
*WSCSTCONT80
*WSCSTCONT132
*WSCSTNONE
*WSCST