X9Ware LLC

# White Paper
# On
# x9.37 File Editing

Revision Date: 08/25/2022
Release R4.10


X9Ware LLC
10753 Indian Head Industrial Blvd
St Louis, Missouri 63132-1101
(844) 937-1850

Email: sales@x9ware.com

## Overview

This white paper explores the ability to edit X9.37 files using generalized and standard file editors that exist within today's desktop environment. At first blush this seems out of the question, due to the very unique format that is used to implement the x9 file format. However, given some reasonable boundaries and limitations, it is more than possible to define and build an environment that will allow you to open, browse, modify, and save x9 files.

This process can be extremely valuable in development and testing environments to build x9 files that are otherwise difficult to create. The ways that you can use such a tool are essentially only limited by your creativity. Several examples are as follows:

1) Create duplicated items for specific use cases.
2) Create various errant file conditions (duplicated headers, duplicated trailers, etc).
3) Rearrange the sequence of records within a bundle or cash letter.
4) Cut and paste items from one x9 file to another.
5) Move a type 61 credit record from one position to another within a file.
6) Insert a type 61 credit into an x9 file when missing.

This x9 file editing process represents an important tool for your x9 toolbox and opens new possibilities and test cases that can now be created.

## Background

The X9.37 specification defines a data format that is used by financial institutions and third party processors for the electronic exchange of check and image data. The file format is unique in the both how the data records are defined and how images are incorporated into the file structure. As in many file structures, there is some history around the layout of these files. The X9.37 file was defined in 2003 around a common interchange format that was previously used for the forwarding and truncation of paper items. This format was then enhanced to allow check images to be incorporated into the file design.

This standard has evolved over time, with the first definition being the DSTU X9.37-2003, more commonly referred to as DSTU. Subsequent standards include:

ANS X9.100-180-2006.

ANS X9.100_187-2008

ANS X9.100_187-2013

Universal Companion Document (UCD) Version 1 applied to ANS X9.100-187-2008

CPA Standard 015 (Canadian Companion Document) applied to ANS X9.100-187-2008

## File Structure

X9.37 files contain a series of record types which are used to build their overall structure. Each file begins with a file header (type 01) and ends with a file trailer (99). Each file can contain one or more cash letters, which begin with a cash letter header (type 10) and end with a cash letter trailer (type 90). Each cash letter can contain one or more bundles, each of which begins with a bundle header (type 20) and ends with a bundle trailer (record type 70).

X9.37 files can be used for either forward presentment or for returned item presentment. In support of these file types, items with each bundle are then appropriately defined as check detail (record type 25) or return item detail (record type 31).

Each item will have a variable number of attached addendum records followed by the image records. The addendum records are used to define the path that an item has taken during its journey from capture through various financial institutions, which are termed the endorsement records. This terminology goes back to the physical endorsements that were applied to the back of checks when processed and cleared on a paper basis. Forward presentment items can include a BOFD endorsement addendum (type 26), an archive addendum (type 27), or multiple secondary endorsement addendums (type 28). Return items can include a BOFD endorsement addendum (type 32), an ancillary data addendum (type 33), an archive addendum (type 34), or multiple secondary endorsement addendums (type 35).

Each item will normally always include their associated images. The image data must be captured in TIFF format using tags as outlined by the X9.100-181 standard. The image segment data must be compressed using the Group4 Fax standard, and each image segment must include a trailing end of facsimile block (EOFB) bit string. Each image is represented by an image view data record (type 50) followed by an image view detail record (type 52). Hence an item will have four image records which represent its associated front and back images (types 50, 52, 50 and 52).

## Character Set Encoding

By convention, X9.37 are most typically encoded using the EBCDIC character set. This is not absolute, since the standard also supports ASCII encoding. There is no identification within the file as to a which character set has been used. Best practices are for readers to inspect the file header record (type 01) and automatically make this determination based on the data present within that first record. All data within an x9 file must be encoded using the same character set.

X9.37 data records cannot contain the null character (0x00) within their actual data. Obviously nulls can appear within the TIFF images.

X9.37 files are typically encoded using upper case characters (eg, “A” and not “a”). However, the standard allows files to be encoded in a mixture of upper and lower case characters. Readers should accept either and should essentially transform any lower data to their corresponding upper case values.

## Variable Length Prefixes

As noted earlier, the X9.37 file structure evolved from earlier standards that were used for check data forwarding and truncation. These file layouts did not include images. All record types within these standards utilized individual 80 byte data records.

With the addition of the image records, there was a new requirement to support variable length records. This resulted in a convention of inserting a four byte binary record length as a prefix to each data record. This binary field is stored in big endian format (eg, for an 80 byte record the value will be 0x00000050). The record length is typically referred to as field zero, since it immediately precedes each X9.37 record definition. The record length is also sometimes referred to as the Inserted Length Field.

Insertion and use of field zero is not mandatory but is considered to be a best practice, since it greatly simplifies the data parsing process for file readers.

The presence and usage of field zero lengths must be determined by the originating and receiving parties. This includes the actual format used to encode the field zero lengths (eg, big endian versus little endian).

## File Format Transformation

Common tools exist within the industry that can transform an X9.37 file from one format to another. These transformations are important, since they often allow an x9 file to be utilized by an application that would otherwise be unable to support the file format. These transformations can include:

- Conversion from EBCDIC to ASCII or ASCII to EBCDIC
- Insertion or removal of field zero lengths

File transformations becomes important in our next topic, which will explore the use of generalized file editors against X9.37 files. These generalized tools do not understand the X9.37 file format. However, they can be coerced into browsing and editing X9.37 files when they can be sensitive to the four byte field zero lengths, which very conveniently define the beginning of each X9.37 record that occurs within the overall x9 file.

Why is this important? It becomes especially important if you want to use a generalized editor to open and then modify the data within an x9 file. Editing will require the presence of field zero within the file. If your x9 file does not contain field zero lengths, then you will have to use a tool to transform and save your x9 file into a format that includes field zero before you use the editor.

Similarly, editors must understand the character set being used to represent the actual data so it can be visually displayed. The easiest way to do this is to create an editing environment that supports either EBCDIC or ASCII data. Alternatively, you can transform your x9 file to a standard character set (eg, EBCDIC) before you begin your editing process.

## Editor Requirements

Most generalized editors support profiles which are activated based on the file extension. This is a common practice to allow the editor to customize their user interaction based on the type of data that is being edited. For example, the editor may adjust its user interface in a certain manner when editing an HTML file, and in a very different way than when editing a “C” source code file.

These profile definitions are the key to the ability to support logical editing of X9.37 files. There are several important capabilities that must be supported by the tool:

- The profile must allow you to define the presence of the field zero (variable length) prefixes that exist between each x9 data record. There is a precedence for this, since various file formats include variable length prefixes. An example is the IBM variable length record definition, which has a variable length indicator in the format 0xLLLL0000 (note that the maximum supported record size is 32k). In our case, we need the ability to indicate that variable length prefixes exist that they are in big endian format (0xLLLLLLLL).

- The profile must allow you to define the character set being used (EBCDIC vs ASCII).

- Depending on the editor, the profile must allow you to define the action that is taken for binary data that is encountered within the file. This is important since the TIFF images will contain long strings of binary data which represent the encoded images. The editor cannot change this data in any way.

- Finally, the editor must be able to open, modify and then write very large files. An x9 file is not your typical file given the very large size that results from the insertion of image records. For example, an x9 file with 20,000 checks may be 250MB in size with 150k to 200k individual records. This very large file size places unique demands on the editor in terms of how data is internally stored and managed.

## A Case Study

As mentioned, this white paper is all about showing you how to use a generalized editor to browse, modify, and save your x9 data files. Hence we need to identify and use an editor as an example to more readily show you how these tasks can be accomplished.

You must consider and then select an editor based on the above stated requirements. There are numerous file editors that are available within today's marketplace. Our goal with this white paper is to provide a concrete example of using a general purpose tool to edit your x9 files. To that end, we will provide an example based on usage of SPFLite, which is a ***shareware tool*** with a line editor that mimics the IBM mainframe SPF editor.

There are many similar tools that are available in the marketplace today. We have chosen SPFLite since it is a proven tool, implements a common user interface, has extensive functionality, includes profiles which support the unique requirements to parse x9 files at the record level, and is immediately available for your download and usage. SPFLite is an example and you can pursue others based on your environment and specific requirements.

Our mission is to show you that x9 file editing is possible using tools that are immediately available to you today. You can decide to use SPFLite based on your evaluation. If you do pursue usage of SPFLite, we would recommend that you consider an appropriate donation (via PayPal) based on the value that you receive from this product. Registration will also allow you to obtain product support and information regarding updates as they become available.

Please be aware that the ability to edit x9 files require SPFLite Build 7.1.4050 or higher. This build level is needed to support the x9 field zero (length) separators in big endian format. There are many earlier versions of SPFLite on various shareware download sites, so you need to double check that you have a build level that contains the needed capabilities.

## Editor Profiles

Most editors will require that a profile be created for each file extension to be updated. For example, if your organization processes x9 files with extensions of .x9 and .x937, then you will define a logical profile which supports each of those. You may require more based on your environment.

A sample x9 profile for SPFLite is as follows:

```
[File]
ProfLock=0
ProfUsing=
AutoBkup=0
AutoSave=3
StateFlag=0
FoldFlag=0
Collate=EBCDIC          Identifies your character set
Source=EBCDIC
Case=T
Preserve=1              Suppresses trailing blank removal
TabsFlag=0
MarkFlag=1
CapsFlag=0
HexMFlag=1
HiFind=1
```

```
HiAuto=1
ImportTabs=0
UndoLevels=0
MINLEN=0
LRECL=0
RECFM=VLI
EOLFlag=NONE
Change=D
ColsFlag=0
ScrollAmount=CSR
Page=0
PageOffset=0
SubArg=
SubCmd=
Start=FIRST
WORD=A-Z a-z 0-9
TABS=""
MARK=""
MASK=""
BNDS="< >>"
```

**Indicates that tabs are not to be expanded**
**Indicates no UNDO needed**

**Indicates that x9 record prefixes are present**
**Indicates no physical line delimiters**

## *Opening an X9 File within the Editor*

Given that background, let's go ahead and open an x9 file using the SPFLite editor.

If by chance you are familiar with the SPF editor under IBM TSO, you will welcome the user interface since it dramatically mimics that user interface. You will be up and running immediately.

SPFLite has an extensive help facility which can be invoked either by the Help command or by hitting the F1 key. It is recommended that you spend some time reading through that material, which will allow you to fully understand that editor functions.

As you will see, SPFLite is a full screen editor that works especially well with block commands. For x9 files, these are ***very powerful functions*** since they allow you to very easily move, copy, replicate, and delete entire blocks of records using one or two commands.

### Primary Commands

Primary commands are entered on the Command line at the top of the screen. They typically affect multiple lines in the file being edited. Multiple commands may be entered at one time on the command line, separated by a command separator character. As in ISPF, the default for this character is the ; semicolon, but this can be altered in the Options - General options window. When the character assigned as the command separator is enclosed in quotes, it is treated as ordinary data.

If the command line is prefixed with an & character, then the command line will be retained in place following command execution rather than being cleared. This allows you to perform the command again by just pressing Enter.

Primary commands are used for many purposes:

- Scroll to a specific line number or Line Label.
- Find a specific line that contains (or does not contain) a search string

- Find and change a character string
- Save the edited data or cancel without saving
- Create or replace other files than the one being edited
- Sort data
- Delete lines
- Undo changes made, subject to the number of maintained Undo levels
- Initiate parallel editing of lines in Power Typing Mode
- Submit jobs for execution using an external process
- Examine and modify data using external Filter programs
- Execute external programs using CMD

You can now use the LINE primary command to apply line-mode or block-mode line commands to a range of lines.

## Visual indication of a file's modified status

When a file has been modified since initially loaded or last saved, an indication is provided by changing the color of the file name on the file tab. See Options - Screen for choosing these colors. You can choose colors that will best convey the status of each file. For example, you could choose to display modified files with red letters for the file tab, and blue letters for unmodified files, with a light background for each. Then, when you see a file tab in red, you would know the file is modified. It is up to you to select colors that will work for you to achieve your desired result.

There is a second file modification indicator - the word Edit in the lower-left corner of the screen on the status line. In unmodified files, you will just see Edit, while in modified files, you will see Edit *.

## Direct Modification of Text

You may modify text by using the arrow keys to move within the data, and type over or insert text as desired. The Insert key will toggle Insert / Overtype mode as needed. (The current status is always visible as INS or OVR in the Status Bar) You can control the size of the cursor in normal mode and in Insert mode and whether it blinks.

## Command Macros

Command Macros are beyond the scope of this brief introduction. Just be aware that command macros can be invoked from the command line, defined with an extension of .MACRO, and stored in the \SPFLite\ MACROS data folder. If such a file (eg, cmdname.MACRO) exists, then SPFLite will execute the script file. Further details on using command macros is provided in Command Macro Support.

## Edit Commands and Command Key Processing

When text is entered on the command line, and a command key is pressed rather than the ENTER key, SPFLite concatenates the contents of the command line to the definition of the command key. The result is handled as a single, composite command by SPFLite. A "command key" is a key mapped to a primary command. For comparison purposes, in IBM ISPF, a 3270 Enter key or PF key would be treated as a "command key", while data keys, cursor keys and PA keys would not.

For example, when you use a Command key defined as a scroll command (UP, DOWN, LEFT, or RIGHT), the value entered on the command line becomes the operand of the scroll command. Assume you have the PageDn key mapped to the DOWN primary command. Entering H or HALF on the command line and

then pressing the PageDn key results in a composite command of DOWN HALF being issued to scroll the screen down by half its height. Similarly, entering the number 200 on the command line and then pressing the PageDn key results in a composite command DOWN 200 being issued, which will scroll forward 200 lines within the file.

### Line Commands

Line commands affect single lines or block of lines. You enter line commands by typing them in the Line Command field on one or more lines and then pressing Enter. The line command field is represented by a column of 6-digit numbers on the far left side of your display. When you are entering data into new temporary blank lines created by the I line command, the line command field contains 6 quotes. This field is also used to display special lines, such as the ==CHG> flag, which indicates a line which has been altered by a primary CHANGE command. You can use line commands to:

- Insert or delete lines
- Repeat lines
- Rearrange lines or overlay portions of lines
- Split lines apart and join lines together
- Perform text paragraph entry and formatting
- Shift data
- Include or exclude lines from the display
- Control tabs and boundaries for editing
- Alter the text case (upper-case, lower-case, sentence-case and title-case)

## *Editing Files and Issuing Commands*

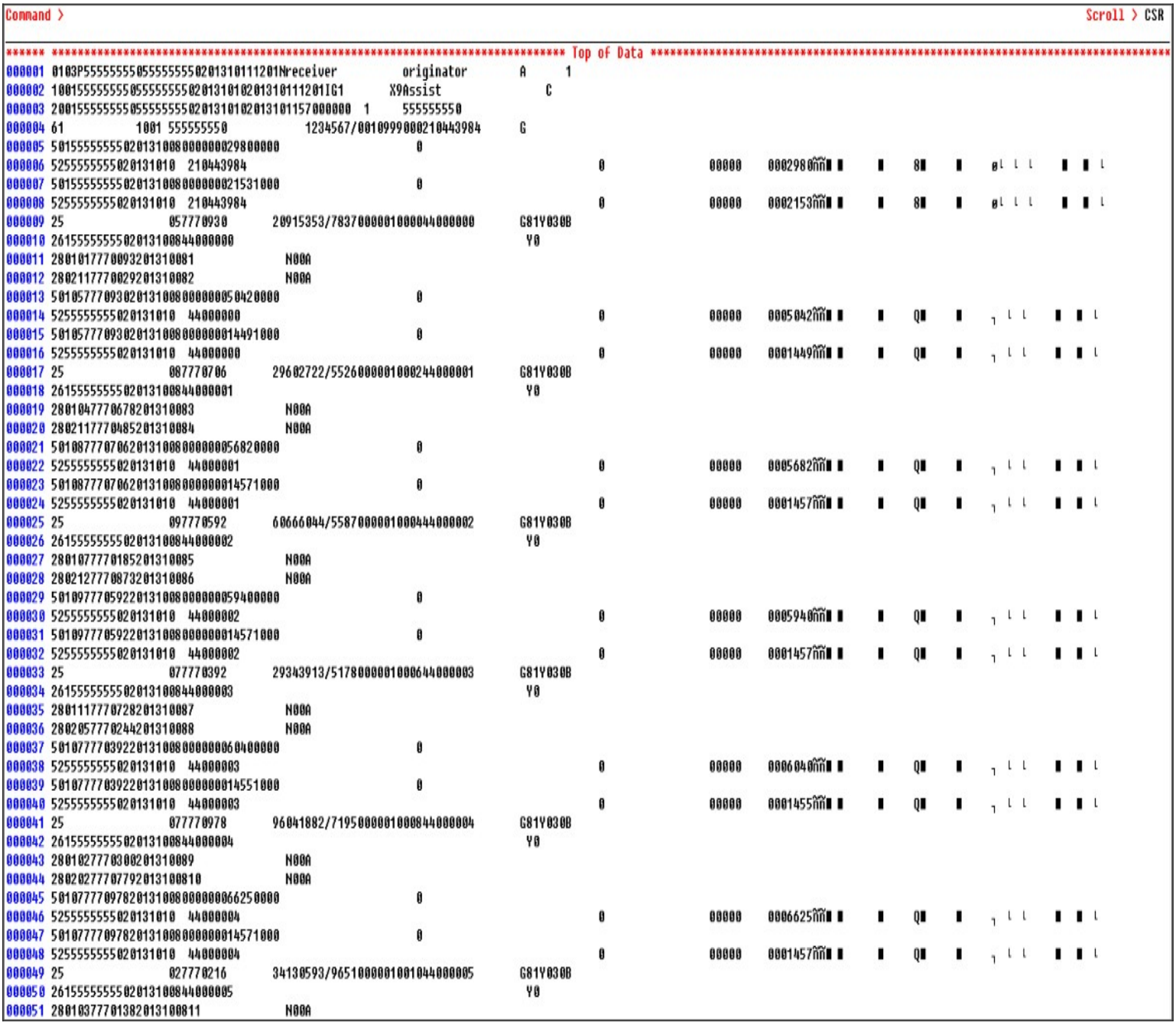
Once you open an x9 file, it will be displayed within the editor in full screen mode. The user interface provides extensive commands to allow you to browse, search, modify, and save the data within the file.

Primary edit commands are entered on the “Command” line after the “>”. Please reference Appendix 1 for a list of all primary commands.

Line commands are entered on individual rows, where you actually enter the line command by over typing the line number itself. Please reference Appendix 2 for a list of all line commands.

## *Sample Editor Screen*

The following screen example has had default colors changed, only to clarify the presentation within this document.




## *File Modification Examples*

Earlier, we indicated that the editor could be used to easily modify an x9 file. Let's now give some high level examples of how that can be easily done to accomplish some otherwise complex file manipulations.

| | | |
|---|---|---|
| 1. | Open a file for browse. | 1. Enter “browse” on the command line.<br>2. Select your file. |
| 2. | Open a file for update. | 1. Enter “edit” on the command line. |

| | | |
|---|---|---|
| | | 2. Select your file. |
| 3. | Open a file for update in a new editor window. | 1. Enter “open” on the command line.<br>2. Select your file. |
| 4. | Save a changed file. | 1. Enter “save” on the command line to overwrite the current file name. Or....<br>2. Enter “saveas” on the command line to save to an alternate new file name that does not currently exist. Or....<br>3. Enter “C/” on the first line (which will copy from this line through the bottom) and then enter “replace” on the command line to replace an existing file. |
| | Find a character string. | 1. Enter “find” on the command line along with the string you want to locate. For example, you would enter “find 12345” or “f 12345”.<br>2. If the search string contains blanks, you must enclose it with the double quote character. For example, you could enter the following: find “search word”. |
| 6. | Delete a single record. | 3. Position to the record to be deleted.<br>4. Use the “D” line command to delete.<br>5. Use the Save primary command to save the file. |
| 7. | Delete a block of records. | 1. Position on the first record to be deleted.<br>2. Use the “DD” line command to mark this first record.<br>3. Position on the last record to be deleted.<br>4. Use the “DD” line command to mark this last record which will delete the entire block that has been marked.<br>5. Use the Save primary command to save the file. |
| 8. | Replicate a single record. | 1. Position on the record to be replicated.<br>2. Use the “R” line command to replicate this record.<br>3. Use the Save primary command to save the file. |
| 9. | Replicate a block of records. | 1. Position on the first record to be replicated.<br>2. Use the “RR” line command to mark this first record.<br>3. Position on the last record to be replicated.<br>4. Use the “RR” line command to mark this last record which will replicate the entire block that has been marked.<br>5. Use the Save primary command to save the file. |
| 10. | Move a single record. | 1. Position to the record to be moved.<br>2. Use the “M” line command to indicate that this record should be moved.<br>3. Position to the target record where the record should be moved to.<br>4. Use the “A” line command to indicate that the identified record should be moved after the current record.<br>5. Use the Save primary command to save the file. |
| 11. | Move a block of records. | 1. Position to the record to be moved.<br>2. Use the “M” line command to indicate that this record should be moved.<br>3. Position to the target record where the record should be moved to.<br>4. Use the “A” line command to indicate that the identified record should be moved after the current record.<br>5. Use the Save primary command to save the file. |

| | | |
|---|---|---|
| 12. | Cut and paste a block of records from one file into another file. | 1. Open the first file.<br>2. Locate the block of records and put “RR' on the first line and “RR on the last line.<br>3. Issue the “replace” command and save this record sequence to a temporary file name of your choice.<br>4. Issue a “cancel” command on the first file.<br>5. Open the second file.<br>6. Locate the position within this file where you want the records inserted and put an “A” on that row.<br>7. Issue the “copy” command and select the temporary file which contains the records to be inserted.<br>8. Save the resulting file. |

## Correcting Trailer Records

After you apply updates to your x9 file, there will most probably be serious issues with your trailer records. This happens when there are changes to the record counts, item counts, dollar amounts, and image counts. These changes will appear in the bundle trailers, cash letter trailers, and the file control trailers. It is almost impossible to correct these trailer records correctly.

The good news is that you can use an x9 tool to automatically repair your trailer records. You will always want to do this after you have made changes to an x9 file using an editor.

## Running X9 Validations

After you apply updates to your x9 file, you will also want to use your x9 tool to run all validations against the file. This is important to ensure that the file still meets all x9 specification requirements. It is also especially true since the file contains binary image data that you can not readily see. Running x9 validation will ensure that the structure of the modified file is correct, including the images, and that all x9 rules are met.

## Summary

The purpose of this white paper is to show that a general purpose editor can be used against x9 files and can create desirable scenarios that are otherwise difficult to impossible to create. You can use an editor to move records within a file, replicate records within a file, delete records within a file, and even cut and paste records across files. All of this is easily accomplished.

Once your file modifications are complete, you can then use x9 tools to automatically validate the modified file and automatically update the bundle, cash letter, and file control trailers. This validation and repair addresses the otherwise complex task of updating the x9 trailer records to ensure that they now match your modified data.

If you have questions about this process, please contact the author at X9Ware LLC.

# Appendix 1: SPFLite Primary Commands

The following is a high level summary of all available primary commands:

Note: You can create your own primary-like commands by using the Command Macro facility. You store a series of primary commands in a file and perform them by typing the Macro name. A Command Macro command is entered on the command line just like a built-in primary command. See Macro Support for more information.

| | |
|---|---|
| ADD | Add a text line |
| APPEND | Add text to end of lines |
| AUTOBKUP | Control backup creation |
| AUTOSAVE | Control automatic file save defaults |
| BOTTOM | Scroll to bottom of file |
| BOUNDS | Set edit boundaries |
| BROWSE | Open a file for browse (read-only) access |
| CANCEL | Cancel edit session without file save |
| CAPS | Set keyboard CAPS mode |
| CASE | Control default literal case handling |
| CHANGE | Change a data string |
| CLIP | Open a new tab using clipboard data |
| CLONE | Open an un-named edit using an existing file |
| CMD | Execute another Program or Command |
| COLLATE | Specify display/sort collating sequence |
| COLS | Control visibility of top Columns line |
| COMPRESS | Compress duplicate strings |
| COPY | Include an external file |
| CREATE | Create an external file |
| CRETRIEV | Conditional Retrieve |
| CUT | Cut data to the clipboard |
| DELETE | Delete selected lines |
| DIR | Display folder containing this file in File Manager |
| DOWN | Scroll downward in the data |
| DROP | Delete selected lines in a Tag group |
| EDIT | Open a file for editing |
| END | End the edit session |
| ENUMWITH | Change Increment for Enumerate Functions |
| EOL | Alter end-of-line mode |
| EXCLUDE | Exclude lines from the display |
| EXIT | Terminate SPFLite session |
| FAVORITE | Add current file to a Favorite list |
| FF | Find in Files |
| FILTER | Process lines with an external filter |
| FIND | Find a character string |
| FLIP | Reverse exclusion status of lines |
| FOLD | Display text in Uppercase only |
| FTP | Start / Stop FTP sessions |
| GLUEWITH | Specify a join string for Glue operations |
| HELP | Display the Help file |
| HEX | Set HEX display mode ON or OFF |
| HIDE | Hide excluded lines |
| HILITE | Control text highlighting options |
| JOIN | Join lines using Find / Change strings |
| KEEP | Delete specific lines in a TAG group |
| KEYMAP | Display keyboard settings dialog |
| LC | Lower-case a range of lines |

| | |
|---|---|
| LEFT | Scroll leftward in the data |
| LOCATE | Scroll the display to a specified line |
| LRECL | Specify record length |
| MAKELIST | Create a FILELIST from the current display in File Manager |
| MARK | Turn MARK lines ON or OFF |
| MEDIT | Add a file to a Multi-Edit session |
| NDELETE | Delete lines where string is not found |
| NFIND | Find lines where string is not found |
| NFLIP | Reverse exclusion status of lines where string is not found |
| NEXCLUDE | Exclude lines where string is not found |
| NREVERT | Revert user line status when string not found |
| NSHOW | Show (unexclude) lines where string is not found |
| NULINE | Mark User line status when string not found |
| OPEN | Edit another file in a new session |
| OPTIONS | Set editor global options |
| PAGE | Set Profile PAGE mode ON or OFF |
| PASTE | Paste data from the clipboard |
| PREPEND | Add text to the beginning of line(s) |
| PRESERVE | Control handling of trailing blanks |
| PRINT | Send selected lines to the printer |
| PROFILE | Display current file profile values |
| PTYPE | Enter PowerType mode |
| QUERY | Display a single Profile setting |
| RCHANGE | Repeat change |
| RECALL | Recall (Open) a favorite FILELIST |
| RECFM | Set record format |
| REDO | Redo an UNDO action |
| RELOAD | Reload current edit file from disk |
| RENAME | Rename the current edit file |
| REPLACE | Replace a file |
| RESET | Reset the display |
| RETF | Recall commands in a forward direction |
| RETRIEVE | Recall previous commands |
| REVERT | Revert User line status |
| RFIND | Repeat the find command |
| RIGHT | Scroll rightward in the data |
| RLOC | Repeat last LOCATE command |
| RLOCFIND | Repeat most recent FIND or LOCATE command |
| RUN | Directly execute the current Edit script |
| SAVE | Save data and continue edit |
| SAVEALL | Save all current tabs |
| SAVEAS | Save data as a new file and switch to it |
| SC | Sentence-case a range of lines |
| SET | Set a command variable |
| SETUNDO | Control UNDO levels |
| SHOW | Show (unexclude) lines where a string is found |
| SORT | Sort the edit data |
| SOURCE | Specify character encoding |
| SPLIT | Split lines using Find / Change strings |
| START | Set initial file position option. |
| STATE | Control edit state saving |
| SUBARG | Set default SUBMIT arguments |
| SUBCMD | Set alternate command for SUBMIT |
| SUBMIT | Pass lines to an external command file |
| SWAP | Switch to Previous or Next Tab |
| TABS | Turn TABS on or off |
| TAG | Alter TAG status of a selection of lines |

| | |
|---|---|
| TC | Title-case a range of lines |
| TOP | Scroll to the top of the file |
| UC | Upper-case a range of lines |
| ULINE | Mark lines as User lines |
| UNDO | Undo changes |
| UP | Scroll upward in the data |
| VSAVE | Perform a virtual save |
| WDIR | Open Windows Explorer for this file folder. |
| XSUBMIT | Submit an external file |
| XTABS | Control handling of incoming tabs |

# Appendix 2: SPFLite Line Commands

The following is a high level summary of all available line commands:

| | |
|---|---|
| A | After destination |
| AA | After block |
| B | Before destination |
| BB | Before block |
| BNDS | Display BOUNDS line |
| C | Copy line(s) |
| CC | Copy a block |
| COLS | Display Columns line |
| D | Delete line(s) |
| DD | Delete a block |
| F | Display first lines in excluded region |
| G | Glue lines together |
| GG | Glue lines together in block |
| H | Here-destination lines |
| HH | Here-destination block |
| I | Insert temporary new line(s) |
| J | Join lines together |
| JJ | Join lines together in block |
| L | Display last lines in excluded region |
| LC | Lower-case lines |
| LCC | Lower-case lines in block |
| M | Move lines |
| MM | Move lines in block |
| MARK | Set column markers |
| MASK | Set the Insert line model |
| MD | Make data lines |
| MN | Make NOTE lines |
| N | Insert permanent new line(s) |
| NOTE / xNOTE | Insert NOTE lines |
| O | Overlay lines |
| OO | Overlay lines in block |
| OR | Overlay-Replace lines |
| ORR | Overlay-Replace lines in block |
| PL | Pad lines |
| PLL | Pad lines in block |
| R | Repeat lines |
| RR | Repeat lines in block |
| S | Show lines |
| SS | Show lines in block |
| SC | Sentence-case lines |
| SCC | Sentence-case lines in block |
| SI | Show indentation |
| T | Select text lines |
| TT | Select text lines in block |
| TABS | Display Tabs line |
| TB | Text Break a line |
| TBB | Text Break lines in block |
| TC | Title-case lines |
| TCC | Title-case lines in block |
| TF | Text-flow a paragraph |
| TFF | Text-flow a block of paragraphs |
| TG | Text glue lines together |

| | |
|---|---|
| TGG | Text glue a block together |
| TJ | Text join lines together |
| TJJ | Text join a block together |
| TL | Trim lines |
| TLL | Trim lines in block |
| TM | Set Text Margin in a paragraph |
| TMM | Set Text Margin in a block of paragraphs |
| TR | Truncate lines |
| TRR | Truncate lines in block |
| TS | Text split a line |
| TT | Text line marking |
| TU | Toggle User Line state of lines |
| TUU | Toggle User Line state of block |
| TX | Toggle excluded state of lines |
| TXX | Toggle excluded state of block |
| UC | Upper-case lines |
| UCC | Upper-case lines in block |
| U | Mark User lines |
| UU | Mark User lines in block |
| V | Revoke User line status |
| VV | Revoke User line status in block |
| W | Swap lines |
| WW | Swap lines in block |
| WORD | Display valid WORD characters |
| X | Exclude lines |
| XX | Exclude lines in block |
| ( | Column shift left |
| (( | Column shift left in block |
| ) | Column shift right |
| )) | Column shift right in block |
| < | Data shift left |
| << | Data shift left in block |
| > | Data shift right |
| >> | Data shift right in block |
| [ | Indent shift left |
| [[ | Indent shift in block |
| ] | Indent shift right |
| ]] | Indent shift right in block |

# Appendix 3: X9Ware LLC

***About X9Ware LLC***

X9Ware LLC provides extensive tools for users of the various x9.37 file specifications. Our product line extends from a free x9 viewer to tools that include validate, modify, repair, make, generate, scrub, import, export, and provide a host of other x9 support and reporting functions. We have price competitive licensing options to meet the needs of any size organization. Our goal is to offer what we believe are some of the best tools in the industry and at the lowest possible cost.

X9Assist is our flagship product which incorporates a series of tools to support your x9 production support, testing, and development requirements. Specific examples are:

| Tool | Usage |
|---|---|
| 1) Make | Creates a new x9 file from a use case file which can be defined in either Excel or CSV format. A variety of fields can be provided via the use case file. Fields that are not present in the use case file can be assigned constant or random data based on your specific requirements. |
| 2) Generate | Creates a new x9 file from a CSV file which can be obtained from various sources. The most typical usage of Generate is as the second step of the Make process (where the output from Make is the input to Generate). |
| 3) Scrub | Obliterates fields within an existing x9 file by replacing values in selected fields with either random or use case data. The x9 data elements that can be altered include MICR line fields, amounts, item sequence numbers, endorsements, and file header records. Scrub also redraws images associated with an altered check after the new MICR fields are assigned, thus allowing the x9 data and the image to remain synchronized. |
| 4) Clone | Clones bundles within existing cash letters to create large x9 files for performance and stress test purposes. |
| 5) Merge | Merges cash letters from multiple independent files into a single, consolidated x9 file that can be used for testing purposes. |

For additional information on the capabilities of X9Assist, please contact lowell.huff@x9ware.com, or visit our website at www.x9ware.com.

# *Appendix 4: SPFLite*

***About SPFLite***

SPFLite is a Shareware software product designed to provide Windows PC users access to an editor which mimics the operation and functionality of the IBM Mainframe ISPF editor. It was developed and is maintained by two former mainframe software developers with over 50 combined years of experience.

As well as duplicating the editing functions of the IBM ISPF editor, SPFLite has added and extended the base ISPF tool-set with many additional functions and facilities. While adhering to the simple ISPF 'green screen' style, it has nonetheless incorporated many optional features using familiar Windows techniques (mouse, common cut/paste keyboard shortcuts etc.)

For additional information on SPFLite, please contact SPFLite@gmail.com, or visit the SPFLite website at www.SPFLite.com.