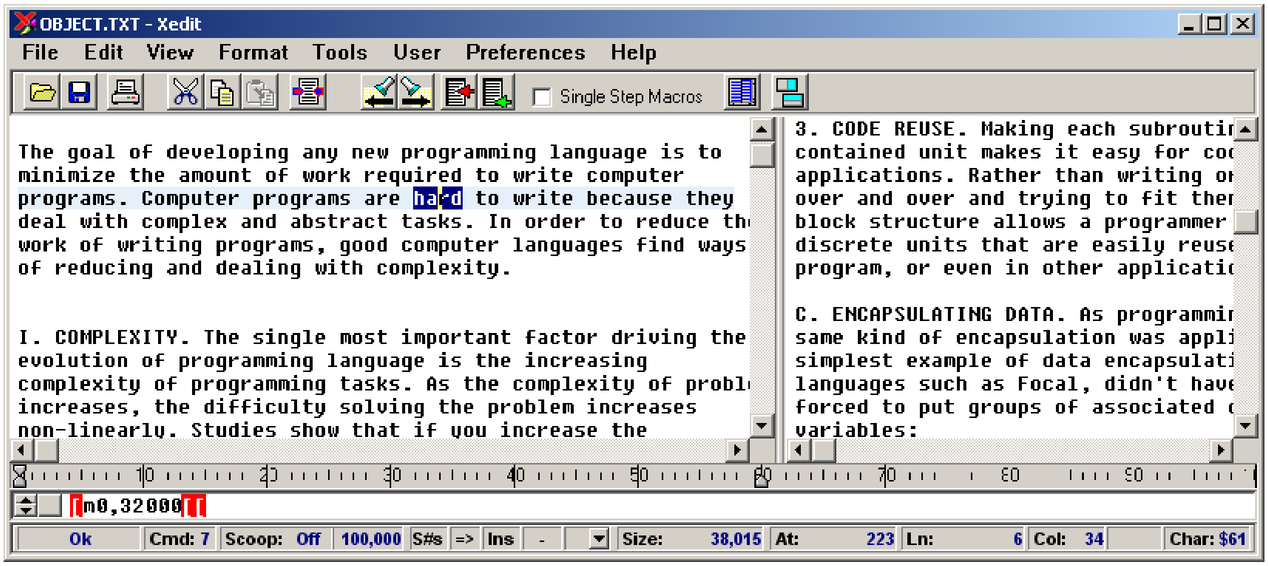
XED For Windows Help



The goal of developing any new programming language is to
minimize the amount of work required to write computer
programs. Computer programs are hard to write because they
deal with complex and abstract tasks. In order to reduce thi
work of writing programs, good computer languages find ways
of reducing and dealing with complexity.

I. COMPLEXITY. The single most important factor driving the
evolution of programming language is the increasing
complexity of programming tasks. As the complexity of probl
increases, the difficulty solving the problem increases
non-linearly. Studies show that if you increase the

3. CODE REUSE. Making each subroutir
contained unit makes it easy for coc
applications. Rather than writing or
over and over and trying to fit ther
block structure allows a programmer
discrete units that are easily reuse
program, or even in other applicatic

C. ENCAPSULATING DATA. As programmir
same kind of encapsulation was appli
simplest example of data encapsulati
languages such as Focal, didn't have
forced to put groups of associated (
variables:

```
m0,32000
```

Ok | Cmd: 7 | Scoop: Off | 100,000 | S#s | => | Ins | - | Size: 38,015 | At: 223 | Ln: 6 | Col: 34 | Char: $61

XED For Windows Help

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Printed: August 2010

Table of Contents

Part I

1 Index

XED is a specialized text editor designed for editing raw ASCII text. Generally speaking, it is not intended for the creation of type-set documents and publishing. It was designed to edit plain-text documents and to give the user complete access to the individual ASCII characters including Control Characters and Extended Characters.

The typical uses for XED include writing and editing source code for programming languages such as Assembly Language, C, Java, Pascal, XPL, etc. It can also be used to edit raw HTML, and XML as well as various kinds of configuration files such as INI, INF etc. One common use for XED is as a replacement for Windows Notepad.

XED has a long history. For a detailed description of the origins of XED, click here.

1.1 Overview of XED

XED is an editor specifically designed for writing programs, but it’s also useful for writing any kind of text. XED is not a word processor and it does not have fancy word processing features. Instead, it concentrates on being fast, powerful and efficient.

XED has features not found in other editors. It can handle files of any size and lines of any length. It handles all ASCII characters including control and extended characters. Although it is not a word processor, it does have features that perform some text formatting tasks such as automatic word-wrap, re-margining paragraphs and centering lines. It has a sophisticated macro capability that lets you to write small programs to do complex editing tasks. It does cut, copy and paste operations – both in line and column oriented modes. If you are using an OEM character set, you can draw boxes and panels using the box character set. It has 32,000 levels of “Undo” and “Redo” so you can restore any text you accidentally delete or restore the text any previous state.
XED also does many kinds of searches: forward and backward, case sensitive and insensitive, wildcards, search, and replace. It has place markers that let you jump back to a marked location in the text or alternate between two marked locations.

XED is fast and efficient. Most commands can be executed with a single keystroke (such as Ctrl+A). Common commands are done without moving your fingers from the home-row key position. In fact, the most common commands are done with the left hand, leaving the right hand free to shuffle papers or scribble notes.

1.2 XED History

XED History. XED has a long and interesting history. Here is a brief story of the development and creation of XED:

VED. The original version of XED was called VED and was written by Peter Boyle around 1977 for Digital Group, 6502-based computers. At the time, most text editors were line-oriented programs that were designed to work on devices like Teletypes and Dec Writers. Since these devices were slow and printed on paper, it could take several minutes to display a full page of text. As a result, the editors of that era were line oriented and only printed a line or two of text at a time.

Peter’s idea was to use the video display of the Digital Group computer to display a full page of text on the screen. A cursor would then be placed at the “editing point.” The text would then scroll around the cursor so the user could always see the full context of his/her entries and changes. This type of editor is so common place these days that it hardly seems remarkable, but it was a radical idea back then.

Command Key Layout. There were several other interesting features of VED. First, it used a very logical command key layout that made it easy to move around through the text. Using the control-key and the letters Q, A, W, S, E and D you could easily move a character, a line or the whole text in any direction. Since these key are immediately adjacent to the Control Key, it is possible to navigate a document with one hand.

Command Line. Another innovative feature of Peter’s editor was a special Command Line that sat at the bottom of the edit screen. With the Command Line, you could enter special macro commands that could perform complex operations on the text. The command line could be re-executed at any point by hitting Control-G. This enabled you to perform repeated operations rapidly and yet it allowed tight control of where and when to execute the macro.

Scoop Buffer. The Scoop Buffer allowed you to grab blocks of text and copy or move any where in the editor. This was long before the invention of the Clipboard.

MR. ED And More. Peter wrote his editor in 6502 assembly code and it was eventually ported to the Apple and several other computers. The idea was so successful that several variations were created by other people in the Denver 6502 community. Probably the most notable version was called “Mr. ED”. It incorporated many advanced feature including a “Teco-like” command structure and multiple Scoop Buffers.

XED. As people gradually migrated away from the 6502 processor, there was a need for a portable version of the editor. As a result, Rich Ottosen wrote the first non-assembly language version of the Editor. It was called XED and it was written in a high-level programming language called XPL. Finally, this version was ported to IBM PC style computers by Loren Blaney and Larry Fish.

Windows Versions. In 1997, Larry Fish created the first Windows version of XED. The first version was created in Delphi-1 and ran under Windows 3.1. This version used 16-bit data structures for most operations but had a 32-bit Edit Buffer so it could work with large blocks of text. Later, a 32-bit version of XED was written under Delphi-3 to take advantage of the 32-bit environment provided by Windows-95. The current version of XED is written in Delphi-6 and has many advanced features. The Windows version of the editor retains many of the original features of earlier versions, but it is truly a Windows program taking full advantage of Windows capabilities and paradigms.

1.3 What's New

This new version of XED represents a complete rewrite of XED. As such, the name has been changed to "XEdit" to differentiate it the last version "Xed32." Nearly every line of the code has been rewritten and there are lots of new features and fundamental changes to underlying functioning of the editor. The main motivation was to fix some basic flaws with the editor. (For a technical description of structural, programming and algorithmic changes to the Editor, please see the "Technical Changes" sections of the “WhatsNew” document that accompanies this installation.)

Here is a list of the most important changes to Editor.

1. Independent Cursor, Highlight And Scrolling. The cursor, highlighting and scrolling are now completely independent. The scroll bars only scroll the screen, they do not move the cursor nor change the highlight. This is more in keeping with standard editors and it makes it easier to copy and move large blocks of text.

2. Buffer Size. By completely reorganizing the way the buffer works, the buffer now automatically and instantly expands and contracts so it never contains even one byte more memory than it needs to hold the text. File size is only limited by virtual memory and disk space. Because of the way the pointer structure works, in theory, it could hold 2.1E12 characters. In real-world tests, I have loaded and edited a 350-megabyte file with no problems or slowing.

3. Multi-View Editor. The Editor now has a feature that allows you to the split the screen to show two views of the same file. Each window is independent and can be scrolled to different parts of the text. At the same time, the two windows are kept in perfect synchrony and any change to one is instantly displayed in the other.

This makes it easy to move text between different parts of the file without losing your place. It also allows you to look at one part of the file while you are editing another. This is ideal for programmers where you want to look at the subroutine parameter while typing the subroutine call.

4. Drag And Drop. The editor now allows you to highlight text and then drag it to another location. Also, holding down the control-key as you drag copies the highlighted text without deleting the source text. Drag and Drop is also allowed between views when the screen is split. These features are consistent with most GUI editors and make it easy to perform certain operations.

5. Columnized Marking. The DOS and Windows versions of XED used the Control-Shift-R command to "scoop" blocks of text along column boundaries. The new version actually highlights the exact columns to show you where the text will be copied/cut.

In addition, the mouse-highlighting and the Cut, Copy and Paste operations have been brought into conformity with the scoop command. Holding down the Alt key while highlighting with the mouse causes the highlight to track columns and not track the raster as it usually does. Likewise, when the ALT key is down,the Copy and Cut options sends columnized blocks of text to the clipboard. Finally, holding down the Alt Key while you paste, inserts text columnized blocks of text by moving or overwriting text as necessary.

6. Macro Debugger. There is now a debugger that can be used to trouble shoot Macro Commands. When the debugger is on, the program displays a debugging window that allows you to single step through the macro one command at a time, so you see the effects of each step on the text. The debugger also displays an assortment of relevant information including, Nesting Level, Command Index, and Loop Count. Finally, the Debugger decodes each command and displays a written description of it function. This saves you having to lookup a command to figure out what it does.

7. Command Line Undo/Redo. Control-O and Shift-Control-O work for Undoing and Re-doing on the Command Line the same as for the main edit screen. The Command Line keeps Undo/Redo information until the you shift to another Command Line. If you hit Control-X, deleting the whole command line and exiting back to the Editor, the Command Line will retain the deleted text. Clicking on on the Command Line and hitting Control-O will restore the text.

8. File I/O. Since the file size is so large, it no longer makes sense to keep an Input and Output file open and shuffle text between the two. Instead, the program simply reads the whole input file into memory and closes it. When you are ready to save the contents, it renames the old input file to "BAK" and then saves the buffer contents to the original filename. No temporary files are necessary.

This means that several Disk I/O COMMANDS are rendered obsolete: "A", "G", "N", "W", "Control-Page Down". "F4".

9. Working Line Shadow. There is now the option to put a light highlight on the line where the cursor is. This makes it easy to see what is on a long line with a lot of tab and spaces. The option can be disabled and you have control over the color.

10. Paragraph Marking. If you double click in the "gutter" along the left side, the nearest paragraph will be highlighted.

11. Toggle Case. The Control-Z command that toggles the case of a characters has an additional feature. If you highlight a block of text, hitting Control-Z will toggle the case of all characters in the highlighted text.

12. File Tools. There is now a specialized tools for copying and merging files. The tool allows you to select any number of input files from diverse directories and combine them into a single output file. The order of the files can be manipulated so the files combined in exactly the way you want. This is much less tedious than the DOS "Copy" command because it requires no typing and you aren't limited by the length of the command line.

13. Copy, Copy And Paste. The Cut, Copy and Paste buttons as well the corresponding menu items work on which ever editor has the focus. This means that you can highlight text on the Command Line and cut/copy/paste with the corresponding buttons. This is a little quicker and easier than using the "Right-click" Context Menu and a lot easier than using the Alt-V command. Hopefully in the future we can get rid of the Ctrl-V/V commands that have been used move text between the line in the past.

14. Remargining. In addition to the normal remargin command, you can highlight a section of text and only remargin that section of text.

15. Code Size. In spite of the fact that there are many new features, the actual code and program size is 23K smaller than the last version. This is result of significant of converting many routines to Object Oriented concepts.

16. Speed. Because of the memory reorganization and other optimizations, the editor is considerably faster, particularly dealing with large files. For example moving from one location to another is instantaneous, because you don't have to move any text to move the cursor. Likewise, Searches are now about 10 times faster because of an improved code structure and because the editor does not need to move characters every time it moves the cursor. Also, unlike the old editor, extremely long lines can be displayed with no noticeable slow down. I have tested the features on single lines of more than 1-million characters and worked just as fast as an 20-character line. Finally, unlike the old editor, line numbers display instantaneously and the line counter does not have to be put in a separate thread.

17. Status Display. Since the buffer size is now virtually unlimited, the "REM" display on the Status Line is no longer useful and has been removed. In its place, the program displays the total number of characters in the file.

18. Fixed Scroll. Because of the independent cursor and scrolling, the Editor no longer supports "fixed-line" scrolling.

19. Scoop Indicator. The Scoop Feature now has its own independent status display. In the past the Scoop used the general purpose status display. Unfortunately, this display would get overwritten by other short-term status conditions, even when the Scoop was still enabled.]

20. Spaces Per Tab Option. The editor now has an option that allows you to set the number spaces that are used for each tab. The value applies to all tabs including printing and conversion tools.

1.4 Getting Started

To run XED, you simply click on its icon in the Windows “Start” menu, “Program Group” or “Desk Top shortcut. You can also run XED by associating it with a particular file extension. Then when you double click on one of the associated files, XED will openthe file and run. (For more information about “File Associations,” see the Windows Help files.)

Here is other important information to help you get started:

The Edit Window
Keyboard Commands

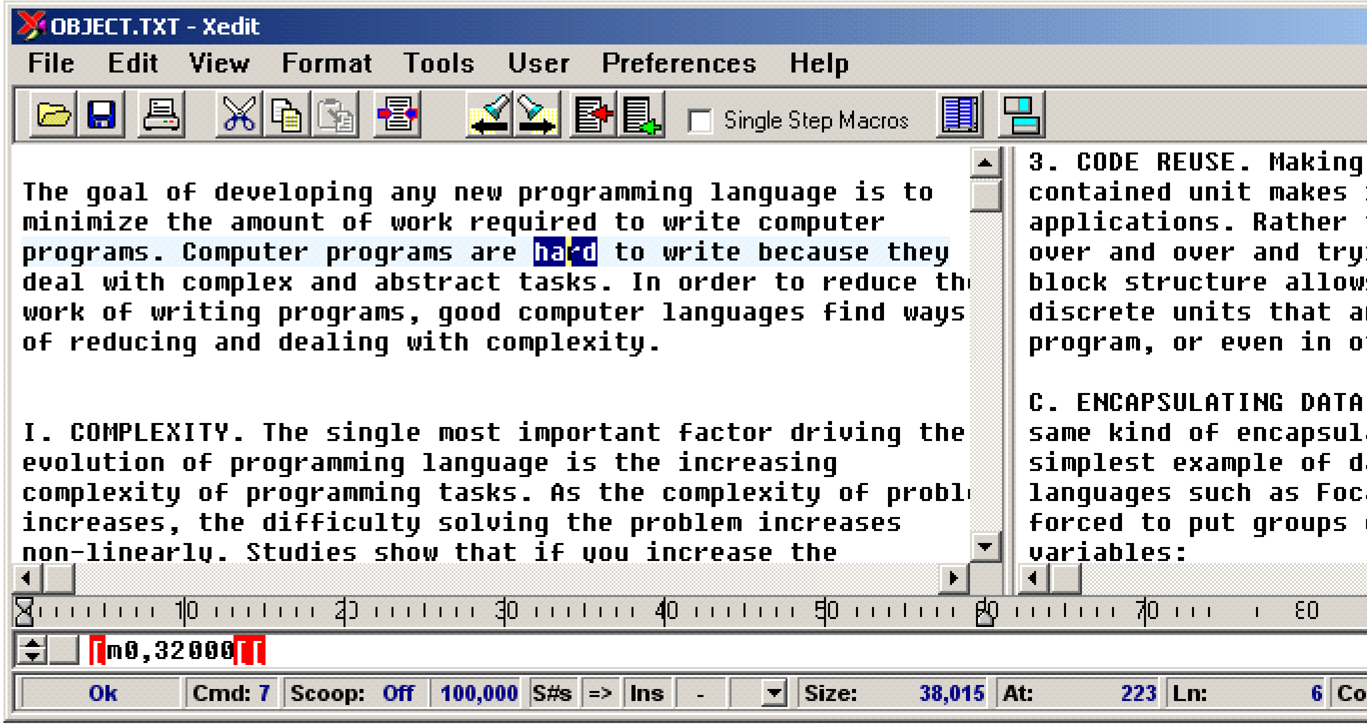
1.5 Basic Operations

The follow links give you detailed information about the basic operations that you can carry out on a document:

The XED Screen
The Edit Window
The Text Ruler
The Line
The Status Line
Secondary Editor
Highlighting Text
Cut, Copy and Paste
Control Characters
Basic File Operations
Undo/Redo Commands
Using the Printer

Deleting
Marking Locations
Margins
Indent Matching
Box Characters
Insert And Overwrite Modes
Long Lines
Scoop Buffer

1.5.1 The XED Screen



```
The goal of developing any new programming language is to
minimize the amount of work required to write computer
programs. Computer programs are hard to write because they
deal with complex and abstract tasks. In order to reduce th
work of writing programs, good computer languages find ways
of reducing and dealing with complexity.


I. COMPLEXITY. The single most important factor driving the
evolution of programming language is the increasing
complexity of programming tasks. As the complexity of probl
increases, the difficulty solving the problem increases
non-linearly. Studies show that if you increase the
```

```
3. CODE REUSE. Making
contained unit makes
applications. Rather
over and over and try
block structure allow
discrete units that a
program, or even in o

C. ENCAPSULATING DATA
same kind of encapsul
simplest example of d
languages such as Foc
forced to put groups
variables:
```

```
m0,32000
```

```
Ok | Cmd: 7 | Scoop: Off | 100,000 | S#s | => | Ins | - | Size: 38,015 | At: 223 | Ln: 6 | Co
```

The XED screen consists of six parts: the Edit Screen, Command Line, Scroll Bars, Ruler, the Tool Bar, Secondary Editor and Status Line. Here is complete description of each.

The XED Screen
The Text Ruler
The Command Line
The Status Line
Secondary Editor

1.5.1.1 The Edit Window

The Edit Window is the large central portion of the window where text is entered. As with most Windows editors, text can be entered only when there is an active cursor on the screen. You can place the active cursor on the edit screen by moving the mouse cursor to the screen and clicking the left mouse button. You can move the cursor to another location by clicking on another location on the screen.

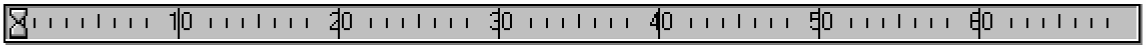
Text is entered by simply typing and the characters will appear at the cursor. If you make a mistake, correct it with the Backspace key. Use the arrow keys to move anywhere in your text. New typing is inserted by shifting existing characters out of the way.

The screen can be scrolled both horizontally and vertically, to keep text you are working on always within the window. The editor responds automatically to the size of the window, so that it is possible edit in small windows, keeping the relevant text in view.

You can highlight text in a variety of ways, using the mouse, the shift key and various cursor movements. You can even highlight individual words by double clicking on them. Highlighted text can be overwritten, or copied to the Windows clipboard.Control Characters are also displayed on the screen as highlighted characters.

Another feature of the screen is popup menus. If you right mouse button click on the main XED Screen, a menu will appear that allows you to Cut, Copy and Paste between XED and the Windows Clipboard.

1.5.1.2 The Text Ruler



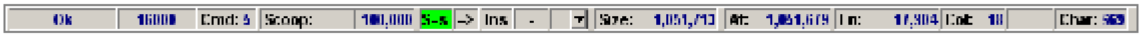
Text Ruler

The Ruler appears at the bottom of the screen and it shows a series of “tick” marks and numbers. These numbers show your column position in the text. If you scroll horizontally through a long line, the ruler will also scroll always showing the proper column. You can left-mouse click on the ruler and it will draw a vertical dotted line through the text. This makes it easy to find the column position of text on the screen.

The Ruler also shows the position of the Left Margin, Right Margin and Margin Indent. These positions are signified by small pointers drawn on the ruler. You can drag these pointers to new positions on the ruler, thereby setting new margin positions. Margins can also be set using the M commands.

If you Right Mouse Button click on the Ruler, a menu of four margin settings will appear. This allows you to quickly change margins settings.

1.5.1.3 The Status Line



The status line displays information about the state of XED. There are 12 fields.

1. Error Field. This field displays error information. If the last operation caused an error, a message appears in this field. If there is no error then “ok” is displayed.

2. Information Field. This field displays information generated by macro commands. For example, the BS and BN commands display the contents of the global variable to this

3. Command Line Number. This field displays the number of the selected command line (0-9).

4. Scoop Enabled. This item indicates whether Scoop Register has been enabled.

5. Scoop Free Space. This field displays the number of unused characters available in the scoop. The scoop is a temporary holding place. It is used to scoop up text and move it to another location. It is similar to what other editors call a clipboard, used for cut, copy, and paste operations.

6. Search Case. This field indicates whether search commands are case sensitive. If “S#s” is displayed, searches are case sensitive. If “S=s” is displayed, upper and lower case characters are treated as the same. You can change the case sensitivity by mouse clicking on it or by typing Control-].

7. Search Direction. This field indicates the direction of searches. If a right arrow appears, searches move forward from the cursor. If a left arrow appears, searches move backward. You can change the search direction by mouse clicking on it or by typing Control-_.

8. Insert/Overwrite Mode. Indicates whether typing is inserted by shifting existing text out of the way or whether the existing text is overwritten. XED is normally used in insert mode. When in insert mode, “ins” appears. When in overwrite mode, “OVR” appears and the color changes to green. You change this value by hitting the Insert key or by mouse clicking on the status item.

9. Wrap Mode. This indicates if the Editor is in Wrap mode. In wrap mode, lines are automatically wrapped as you type. You can toggle this feature by hitting F3 or by clicking on the status item.

10. Block Character Mode. This displays whether or not you can enter box characters from the numeric keypad. It also displays the type of box characters that you are using. You can cycle through the different box character types by typing Control-Shift-D or by clicking on the Arrow button next to the item.

11. Size. This field indicates the current size of the file. As the you type and insert characters into the editor, this number will increase..

12. At. This field shows the cursor location, in characters, from the beginning of the edit buffer.

13. Ln. This field indicates the Number of the Line that the cursor is on.

14. Col. This field shows the cursor column position from the beginning of the line. The column number takes into account the effect of any tab characters. Note that the left-most column is column zero.

15. Modified. This item signals if the text in the editor has been modified. The Modified Flag is cleared whenever the text is saved.

16. Char. This item shows the hexadecimal ASCII value of the character just before the cursor.

1.5.1.4 Secondary Editor

It is often useful to view different parts of the text at the same time. For example, you might want to view one part of the text while type information into another. For the reason, the Editor allows you to display a Secondary Editor that will show a second copy or view of the file. Each edit window is independent and can be scrolled to different parts of the text. At the same time, the two windows are kept in perfect synchrony and any change to one is instantly displayed in the other. You can also drag and drop text between editors, which makes it easy to copy text between far distant parts of the text.

To activate the Second Editor, press the “Toggle Second Editor” button on the Tool Bar. You can also display the Second Editor by selecting the “View|Display Second Editor” option from the menu bar.

When the Second Editor is displayed, you have the option of splitting the screen horizontally or vertically. To toggle between Horizontal and Vertical modes, press the “Toggle Second Editor Position” button. You can also change the way the screen is split be selecting the “Horizontal” or “Vertical” split options from the “View” menu item.

1.5.2 Mouse And Scoop Highlighting

When you hold down the left mouse button and drag the cursor across the screen, a highlight appears, marking a block of text. The highlighted text can be deleted by hitting the “Delete” key. It can be overwritten, by typing, pasting or inserting text. Finally, it can be copied or cut to the clipboard using one of the menu items or tool bar buttons.

To allow large blocks of text to be highlighted, the program automatically scrolls if the cursor moves off the screen. To enable precise highlighting, the speed of scrolling varies depending on how far the cursor moves off the screen. If the cursor is one line off the screen, the scrolling is slow. If it is greater than one line off the screen, the scrolling is fast.

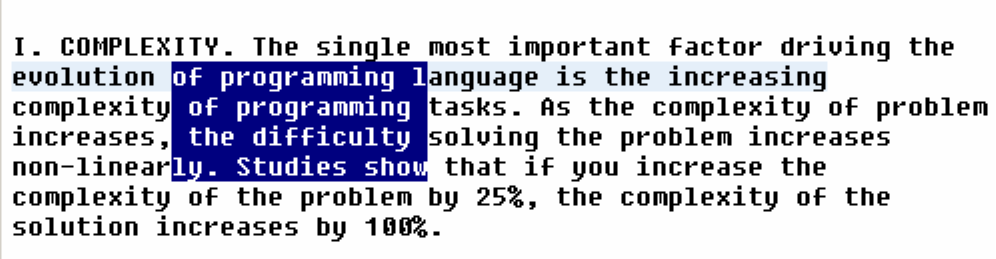
Also, holding down the shift and moving the cursor in any manner causes the intervening text to be highlighted. For example, if you click the mouse on a location in the text, then hold down the shift key and click on another location, the entire text in between will be highlighted. This allows you to accurately highlighting of large blocks of text.

One of the more useful features makes it easy to highlight single words. If you double click on the text, the nearest single word single word will be highlighted.

Drag and Drop. Once a block of text has been highlighted, you can click on it and drag it to another location. Also, holding down the control-key as you drag causes the highlighted text to be copied without deleting the source text. Drag and Drop is also allowed between views when the screen is split.

Columnized Marking, Cutting, Copying and Deleting. It is often useful to grab a block of text along column lines. For example, you might want to scoop up a column of numbers that is embedded in a table without getting other information from the table. Likewise, you may want to insert blocks of texts into a specific column in a table without damaging the table structure. For this reason, XED provides another model for copy and pasting text.

To enable “Column Cut/Copy,” hold down the Alt key before you begin highlighting text, the highlight will follow the columns and mark a rectangular block of text.



```
I. COMPLEXITY. The single most important factor driving the
evolution of programming language is the increasing
complexity of programming tasks. As the complexity of problem
increases, the difficulty solving the problem increases
non-linearly. Studies show that if you increase the
complexity of the problem by 25%, the complexity of the
solution increases by 100%.
```

When a Columnized block of text is highlighted, the marked columns can be copied or cut to the clipboard. If you Cut a Columnized block of text to the Clipboard, the highlighted area will be overwritten with spaces to maintain the original text alignment.

If hit any of the keys that delete, such as Control-X, Backspace or Delete, the block of text will be deleted. If Overwrite Mode is on, the block of text will be overwritten with spaces. If you hit Control-X, the text will be deleted and copied to the clipboard just like the Cut command does.

Pasting Along Columns. Once you have text in the Clipboard, you can Paste it along column boundaries. (This works with any text no matter how it got into the Clipboard.)

To Insert along columns, simply move the cursor to the line and column where you want the text and press the Alt key while you paste. The block of text will be inserted so that each line of block will appear at the same starting column as the cursor, moving or overwriting text as necessary to maintain the original column format. Here is an example:

```
| This is a small piece |
| of text that can be   |
| inserted anywhere.    |
```

```
SEE THE QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE ------------------------QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE | This is a small piece |QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE | of text that can be   |QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE | inserted anywhere.    |QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE -------------------------QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE QUICK BROWN FOX JUMP OVER THE LAZY DOG
```

As you see, the small block of text has been inserted into the middle of the larger block by moving part of each row to the right. If Overwrite mode is enabled, the text would have been overwritten, without moving any text to the right.

1.5.3 Control Characters

When control characters are inserted into the text they have special coloring so they stand out from the rest of the text. For example, in the default color scheme, Ctrl+L appears as a red-on-white “L”. Some control characters, like carriage return and tab, position the text and don’t appear as highlighted characters. Sometimes it is useful to see all of the control characters. Ctrl+T highlights all control characters, including tabs and carriage returns. It also displays space characters as underlines so you can easily count them and distinguish them from tabs.

Occasionally you need to insert a control character, such as the bell (Ctrl+G), into your text. But typing most control characters executes a command rather than inserts it. The Ctrl+^ (same as: Ctrl +Shift+6) is used to get around this problem. Any keystroke that follows this command is inserted into the text.

1.5.4 Reading And Writing Files

XED allows you to open and operate on files in two fundamental ways: XED style and Windows style. The XED style operations use command line commands such as “A”, and “W”. With the exception of “T” and “Q” these commands will not prompt you to save the file if the information has changed. This is based on the idea that these commands are meant to be powerful and primitive. They are described in detail in Deferred Mode section.

The Windows style file operations consist of commands like “New,” “Open,” “Save,” and “Save As” which are found on the menu bar. Here is a summary of their actions:

NEW – This commands closes all files and clears all text from the edit buffer. If there have been any changes, it will prompt you to save them.

OPEN – This command allows you to open an existing file on the disk. It then closes the current file and clears the edit buffer. If there have been any changes, it will prompt you to save them.

SAVE – This command saves the current file to disk, creating a backup. It does this without closing the file or the need to relocate your position in the text. The feature works with multiple buffer files.

SAVE AS – Works the same as the SAVE command except that it allows you to save the test to a different file. The command handles moving backup files and all information to the new path and filename and reopens the file at the same position as the original file.

REOPEN – The REOPEN command displays a list of the last five files opened by XED. To reopen one of these, simply click on one of the files.

1.5.5 Undoing and Redoing

XED has the capability of undoing the last 32,000 changes to the edit buffer. This includes not only inserting and deleting text, but all cursor movements and complex operations like remargining paragraphs. Each undo step can be accomplished by hitting Control-O on the keyboard or selecting “Edit|Undo” on the menu bar.

XED also has the ability “Redo” any series of commands that have been “Undone.” The “redos” must be done immediately after the “undos.” If any other changes are made after the “undo” process, the possibility of redoing is lost. Redoing is accomplished by hitting Control-Shift-O on the keyboard or selecting “Edit|Redo” from the menu bar.

1.5.6 Using The Printer

XED allows you to print your documents to any printer attached to Window. It also allows you to configure the printing margins and other things that effect the appearance of your printout.

Printing. To print a document, select “File|Print” from the menu bar.

Page Setup. The Page Setup Dialog allows you to control the way the text appears on the page. To open the Page Setup Dialog, select “File|Print| from the menu bar. There are three sections in the Setup Dialog:

1. White Space. These options control the amount of white space around the edge page. The setting is displayed in inches.

2. Clip Lines. This option controls whether long lines wrap when reach the right margin or wether they are truncated.

3. Print Page Numbers. When this option is enabled, the program will print page numbers in the upper right corner of every page except the first one.

Print Setup. This option calls the standard Windows setup program from the printers connected to your computer.

1.5.7 Deleting

The Backspace key is often used for correcting mistakes. It deletes one character to the left of the cursor. The Delete key deletes the character to the right of the cursor, which has the effect of deleting one character ahead. Ctrl+X has two modes of operation. If text is highlighted, Ctrl-X follows the Windows standard by “cutting” the text to clipboard. If no text is highlighted, Ctrl-X deletes from the cursor back to the beginning of the line, or if the cursor is at the beginning of the line, it deletes the line above.

1.5.8 Markers

Text markers can be used to move back and forth between two locations. This is ideal when you need to compare two pieces of text.

You can put an invisible marker in the text at any time by hitting the F9 key. You can move to another location and place a second marker by typing F9 again. Only two markers are active at one time. If you hit F9 a third time, the oldest marker is moved to the current position. The Marker not only remembers the cursor position but how the screen is scroll so that the marker returns you to the exact same screen image that was marked. This is useful for comparing two blocks text by flipping back and forth between two portions of text.

If you hit F10, the cursor moves to either the most recently set marker, or if you just moved to a marker, the cursor moves to the other marker.

1.5.9 Margins

XED has several ways to format text. The first is the margin command. This formats a paragraph so that the margins are aligned to a specified position. The “M” margin command can be used with one, two or three arguments. If three arguments are used, the left and right margins and the paragraph indenting are set. For example:

```
<Esc>M5,60,10<Esc><Esc>
```

The first argument is the left margin, the second is the right margin and the third is the number of spaces to indent the first line of a paragraph. The third argument is optional.

The position of the Margins and Paragraph Indent is visually displayed on the<>Text Ruler. They are shown by the position of the small tabs on the Text Ruler. The bottom two tabs show the position of the Margins and the upper tab shows the Indent. You can also set the position of the Margins and Indent by dragging the tabs with the mouse.

Entering the arguments sets the positions; it does not actually remargin a paragraph. Actual remargining is accomplished by entering the “M” command without arguments or by hitting F6.

As an example, if you select margin settings of M5,60,10 then hit F6, the paragraph at the cursor is formatted so that the left margin is five spaces from the left side of the screen, the right margin is at column 60, and the first line of the paragraph is indented 10 spaces. Once you have entered the margin specification, you can format other paragraphs. If you “margined” using the “M” command without arguments, you can margin other paragraphs using Ctrl+G to re-execute the command line. Hitting F6 margins a paragraph without resorting to the command line.

The next text-formatting feature is called word-wrap. It automatically inserts a carriage return at the end of each line as you type it. This saves you from having to hit Enter. Word wrap occurs when you are typing characters and the cursor reaches the right margin. At this point XED searches backward for the first space. If it finds one, it replaces it with a carriage return. If no space is found within 16 characters, no word-wrap occurs until a space is typed. Word-wrap only occurs on space characters.

Word-wrap mode is turned on and off with the F3 key. XED starts with word-wrap turned off. Striking F3 turns it on. Striking F3 again turns it off. When word-wrap is turned on, the up arrow (^) marker on the ruler (which indicates the bell column) is changed to a diamond.

When XED is in word-wrap mode it inserts spaces, if necessary, in front of each new line. This makes the word-wrapping consistent with the margin command.

1.5.10 Indent Matching

Ctrl+U is a Immediate Mode command that is used to indent the current line to the same column as the line above. It is frequently used by programmers to indent blocks of code.

The final text formatting command, Esc-Y, is used to center a line of text. This is useful for titles. Space characters are inserted in front of the line so that it is evenly centered between the left and right margins.

1.5.11 Box Characters

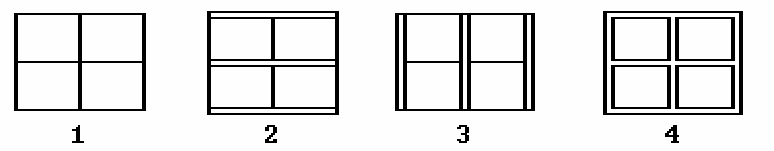
Under DOS, the video displayed used a special type of character set that had special graphic characters. These characters included special “Box Characters that allowed you to draw a variety of boxes on the screen. Windows typically uses fonts that do not have these box characters, but you can still view and manipulate them in XED. To do this, you must select an OEM font such as the

“Terminal” font.

All fonts have extended characters sets beyond the standard alphabetic characters. These characters consist of a variety of symbols including the Box Characters. All these symbols have ASCII values between 128 and 254. These character sets also have symbols defined for control characters (1 through 31). XED can display all of the symbols for the characters from 128 through 254. XED does not display the symbols for control characters because it’s more important to be able to recognize the control character than it is to view these symbols.

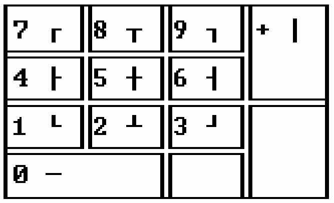
The complete set of ASCII characters can be displayed by typing Alt+F2. This also provides a convenient way to insert an extended ASCII character into your text. Simply type its hex code.

The Box Characters. You can make nice looking charts and drawings using the box characters shown here:



One way to get these extended ASCII characters is to use Alt+F2, but there is an easier way. Press Ctrl-Shift-D key to enter Box Character mode. When you do this, a box character will appear in the Box Character item on the Status Line. You can also put the editor in Box Character Mode by click on the down arrow next to the Box Character item on the Status Line.

When this mode is enabled, the numeric keypad is used to type the box characters as follows:



7 ┌	8 ┬	9 ┐	+ │
4 ├	5 ┼	6 ┤	
1 └	2 ┴	3 ┘	
0 ─			

There are four sets of box characters consisting of combinations of single and double lines. Each time you press Alt+B, you select the next set. Pressing Ctrl-Shift-D five times sequences through the four sets of box characters and back to the normal numeric characters (whereupon the box character disappears from the status line).

1.5.12 Insert And Overwrite Modes

XED has two modes for entering text: insert mode, and overwrite mode. Insert mode is the normal mode. When XED is in this mode, existing characters are shifted to the right before a new character is inserted. This way no text is lost as new text is added. In overwrite mode, the character under the cursor is destroyed whenever a new character is typed.

XED starts in insert mode. Hitting the Insert key changes between insert and overwrite mode. When XED is in insert mode, the letters “ins” appear on the status line, and when it’s in overwrite mode, the letters “OVR” appear on the status line. Because overwrite mode can be dangerous, the “OVR” symbol flashes to alert you.

Overwrite mode is useful for making diagrams and drawing box characters. You can fill an area with spaces then type in the appropriate text. In overwrite mode the text won’t shift.

When in overwrite mode, inserting the scoop overwrites the existing text.This can be useful in some instances, and a disaster in others. XED protects you as much as possible from over-write errors. For example, it does not overwrite a carriage return. Also, it does not overwrite if the character you enter is a carriage return. This way, entering blank lines does not overwrite existing text.

The Backspace key works differently in overwrite mode. In overwrite mode, Backspace replaces the deleted character with a space. This way, the text doesn’t shift. To prevent extraneous spaces from accumulating at the beginning of a line, Backspace does not replace carriage returns with a space—it just deletes them.

1.5.13 Long Lines

Most editors have trouble handling extremely long lines of text. This usually occurs when trying to edit some sort of hex loader file that has no carriage returns. Some editors get completely confused by long lines, while others simply do not display characters past a certain point in the line. XED handles lines of unlimited length without any confusion. A long line appears across the top of the screen as a single line.

The current version of XED for Windows has been optimized for long lines. Since calculating tab positions requires analyzing the whole line, displaying tab positions can be problematic on a very long line. For this reason, if a single is longer than 1024 characters, XED will not expand any tab past this point on that line. The current version has been tested on lines as long as 2-million characters with no noticeable slowing or degradation of performance.

1.5.14 Scoop Buffer

The scoop is used to pick up segments of text and move them to other locations. Words, phrases, paragraphs, and even large blocks of text can be moved easily. The scoop also can be used to duplicate the same block of text over and over. The scoop is similar to the Clipboard in Windows except that it can do certain things that the Clipboard cannot do. For example, when text is added to the clipboard, it always overwrites the old text in the Clipboard buffer. New text cannot be combined with text that is already there. With the Scoop Buffer, text from different parts of the document can be added and combined.

Line Scoops. With XED in Immediate Mode, the scoop is turned on by typing Ctrl+R and turned off with Ctrl+F. When the scoop is on, text can be pulled into it by moving the cursor backward over the text. Any backward movement of the cursor, including deletes, scoops up text. You can delete and scoop at the same time.

While the scoop is on, moving the cursor forward causes the equivalent amount of text to be removed from the scoop. Be careful! Generally, the scoop should be turned off before resuming normal editing.

The text that is in the scoop can be inserted into the edit buffer at any time. Executing an “X” command on the command line or F5 from direct mode causes the entire contents of the scoop to be inserted into the edit buffer at the cursor. The text remains intact in the scoop until it is erased with an “E” on the command line or F8. This allows many copies to be inserted into different locations in the edit buffer.

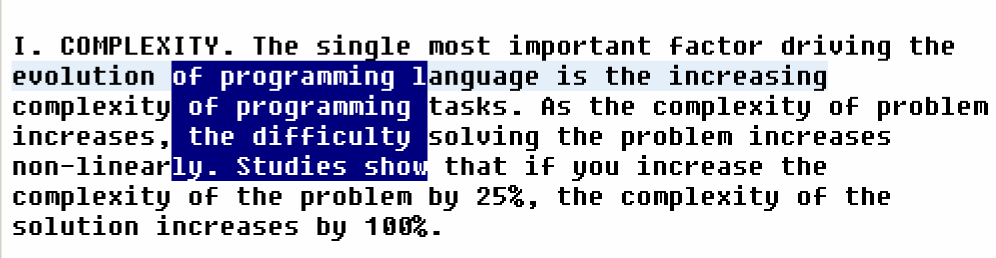
Text from the scoop also can be inserted into the edit buffer using Ctrl+Y. This inserts the text and erases the scoop, which is useful when moving lots of pieces of text.

You can scoop up to 100,000 characters. Since the edit buffer can hold millions of characters, some care must be taken not overflow the Scoop Buffer. The amount of free space available in the Scoop Buffer is displayed on the Status Line.

Column Scoops. It is often useful to grab a block of text along column lines. For example, you might want to scoop up a column of numbers that is embedded in a table without getting other information from the table. Likewise, you may want to insert blocks of texts into a specific column in a table without damaging the table structure. For this reason, XED provides another model for Scooping and Inserting text.

In “Column Mode,” text is “Scooped” along column boundaries. Likewise, in this mode, text can be inserted so that it maintains its column structure so that each line of the inserted text starts at the same column. If some of the lines you are inserting into are shorter than the specified column, then these short lines are extended with spaces to allow each inserted line to start in the same column. Any tabs on the line are also taken into account.

Scooping Columns. To begin a “Column Scoop,” move the cursor to either the lower-right or upper-left corner of the rectangle you want to scoop. Then press Control-Shift-R. This puts the editor in scoop-mode, ready to select a set of columns. Next, you move the cursor to the opposite corner. This will highlight a block of text. (Note that until you move the cursor away from the starting column, no highlight will appear.)



```
I. COMPLEXITY. The single most important factor driving the
evolution of programming language is the increasing
complexity of programming tasks. As the complexity of problem
increases, the difficulty solving the problem increases
non-linearly. Studies show that if you increase the
complexity of the problem by 25%, the complexity of the
solution increases by 100%.
```

When you have highlighted the block of text, press Control-F. This causes the characters in the highlighted text to copies to the scoop buffer. You can also press Control-Shift-F to copy the highlighted and fill the block with spaces. This allows you to maintain the original column alignments, while removing a column of text.

Note that unlike the line scoop command, it doesn’t matter if you start at the lower-right corner or the upper-left corner, as long as you move the cursor to the opposite corner before you press Control-F/ Control-Shift-F. (Do not start at the lower-left or upper-right corners.) You can change the starting cursor position simply by pressing Control-Shift-R at another location.

Inserting Along Columns. Once you have text in the Scoop Buffer, you can insert it along column boundaries. (This works even if you did not Scoop the text along column boundaries.) To do this, simply move the cursor to the line and column where you want the text and press Alt+F5. The block of text will be inserted so that each line of block will appear at the same starting column as the cursor, moving or overwriting text as necessary to maintain the original column format.. Here is an example:

```
| This is a small piece |
| of text that can be   |
| inserted anywhere.    |
```

```
SEE THE QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE -------------------------QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE | This is a small piece |QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE | of text that can be   |QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE | inserted anywhere.    |QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE -------------------------QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE QUICK BROWN FOX JUMP OVER THE LAZY DOG
SEE THE QUICK BROWN FOX JUMP OVER THE LAZY DOG
```

As you can see from the example, the small block of text has been inserted into the middle of the larger block by shifting part of the text to the right. If Overwrite Mode is enabled, the original text will be overwritten without shifting any characters.

1.6 Keyboard Commands

Many of the operation of XED are carried out by typing various combinations of keys on the keyboard. Although XED supports all of the standard Windows mouse operations, using a mouse while editing text is often very clumsy. Therefore XED emphasizes keyboard commands.

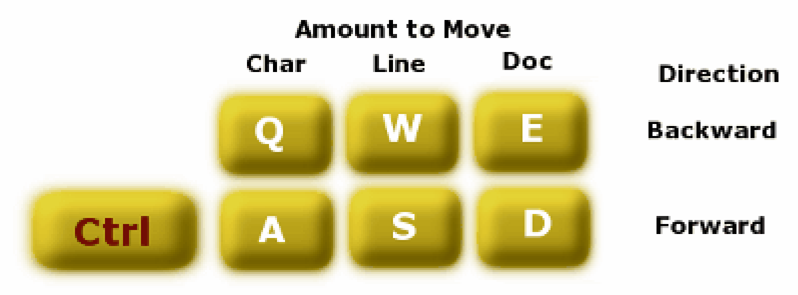
XED has two kinds of keyboard commands: those that operate directly on the text and those that are executed from the Command Line. These two modes of operation are called Immediate Mode and Deferred Mode. XED always starts in Immediate Mode. You can switch to Deferred Mode by typing Escape or just clicking the mouse cursor on the Command Line.

Here is comprehensive list of all the commands:

Immediate Mode
Immediate Commands
Deferred Mode
Deferred Commands
Unimplemented Commands.
Command Line Editor Keys.

1.6.1 Immediate Mode

Most of the commands used in Immediate Mode are control characters. The most commonly used commands are positioned close to the left side of the keyboard so that the Ctrl key can be pressed with the little finger while the commands are typed. This way your fingers never leave the home position. Also, you only need one hand to move through the text. This leaves the other hand free to do other things. The following diagram shows the location of the main cursor movement keys:



Note that the commands are arranged in pairs. One key moves the cursor backward, while the other moves the cursor forward. As you see, the cursor can be moved a character at a time, a line at a time, or to the beginning or end of the edit buffer.

A few years ago IBM (in its infinite wisdom) moved the Caps Lock key to the Ctrl key position, and made the key arrangement shown above less useful. You can still operate the control key with the opposite little finger, but you cannot do one-handed operation. One option to deal with this problem is to use a utility that will swap the Caps Lock and Ctrl Key in software. One such utility is called: Ctrl2Cap and versions for both XP and Win98 are available here:

http://www.sysinternals.com/

Other options include rewiring your keyboard and purchasing a programmable keyboard that allows key remapping.

In XED the line-movement commands are a little different than in most other editors. In most editors the line-movement commands leave the cursor on the same column, but in XED line-movement commands force the cursor to the beginning of the line. This helps orient you as you scroll through your text. If you prefer the cursor to stay on the same column then use the arrow keys.

For a complete list of Immediate Mode Commands, click here.

1.6.1.1 Immediate Commands

Immediate Commands work as soon as the key is struck. They are generally cursor movement commands, text formatting commands and commands that change the way the editor works. The following table lists various categories of Immediate Commands:

Cursor Movement	Command Line Commands
Delete Commands	File Commands
Insert Special Characters	Mode Commands
Formatting Commands	Search Commands
Clipboard/Scoop Commands	Display Commands

Note: The Alt key is extensively by Windows. Usually it is used in combination with the an alphabetic keys to select items from the menu bar. As a result, the Windows version of XED uses the Control-Shift combination in place of Alt for alphabetic commands. Combination that used Alt and a function key, remain unchanged.

1.6.1.1.1 Cursor Movement Commands

The following keystrokes move the cursor around the screen and through the text:

Cursor Movement

Key Stroke	Description	Status
Ctrl-A	Ahead a character	
Ctrl-Q	Back a character	
Ctrl-S	Down a line	
Ctrl-W	Up a line	
Ctrl-J	Jump ahead a page (PgDn)	
Ctrl-K	Kick back a page (PgUp)	
Ctrl-D	Ahead to end of edit buffer	
Ctrl-E	Back to start of edit buffer	
Left Arrow	Back a character	
Right Arrow	Forward a character	
Page Down	Move one page down	
Page Up	Move one page up	
All these command stay on the same column		
Up Arrow	Up a line	(%)
Down Arrow	Down a line	(%)
Ctrl-Up-Arrow	Scroll Up a line	
Ctrl-Down-Arrow	Scroll Down a line	

Word movement commands are work based on XED global settings , which control how the commands deal with the white space between words. Commands with an asterisk work opposite to the global settings.

Word movement commands

Key Stroke	Description	Status

Ctrl-N	Move to next word	
Ctrl-B	Move to last word	
Ctrl-Shft-N	Move to next word (*)	(#)
Ctrl-Shft-B	Move to last word (*)	(#)

# = Indicates that the key sequence has changed in the Windows version.

% **= I**ndicates that the key sequence is new and not available in the DOS version.

1.6.1.1.2 Delete Commands

The following keystrokes delete characters from the text:

Delete

Key Stroke	**Description**	**Status**
Bksp	Delete back a character	
Del	Delete ahead a character	
Ctrl-C	Delete back a word	
Ctrl-Shft-C	Delete back a word	(#)
Ctrl-X	Delete back a line	
Ctrl-Shft-X	Delete ahead a line	(#)
Ctrl-O	Undelete/Undo any change	
Ctrl-Shift-O	Redo	

# = Indicates that the key sequence has changed in the Windows version.

% **= I**ndicates that the key sequence is new and not available in the DOS version.

1.6.1.1.3 Insert Special Characters

The following keystrokes insert special characters into the text:

Insert Special Characters

Key Stroke	**Description**	**Status**
Ctrl-I	Tab	
Ctrl-L	Form feed (new page)	
Ctrl-^	Always insert next keystroke	

# = Indicates that the key sequence has changed in the Windows version.

% **= I**ndicates that the key sequence is new and not available in the DOS version.

1.6.1.1.4 Formating Commands

The following keystrokes reformat the text in various ways:

Formating Commands

Key Stroke	**Description**	**Status**
Ctrl-U	Indent same as line above	

Ctrl-Z	Shift case and move ahead
F6	Margin paragraph

# = Indicates that the key sequence has changed in the Windows version.

% **= I**ndicates that the key sequence is new and not available in the DOS version.

1.6.1.1.5 Scoop Commands

The following keystrokes move text in and out of the Scoop Buffer :

Clipboard

Key Stroke	**Description**	**Status**
Ctrl-X	Cuts highlighted text to clipboard	
Ctrl-V	Pastes the contents of the clipboard	

Scoop

Key Stroke	**Description**	**Status**
Ctrl-R	Start (or Resume) a scoop-up	
Ctrl-F	Finish a scoop-up	
Ctrl-Y	Insert scoop and empty it	
F4	Get next buffer from disk	
F5	Insert scoop	
F8	Erase scoop	
Ctrl-F1	Copy Win Clipboard to Scoop	
Ctrl-F2	Copy Scoop to Win Clipboard	
Alt-F5	Insert column scoop	
Alt-F9	Copies word-search to command line	
Alt-F10	Copies word-search to command line	
Ctrl-Shft-R	Start column scoop-up	(#)
Ctrl-Shft-F	Finish column scoop & blank	(#)
Ctrl-Shft-Y	Insert column scoop & empty	(#)

# = Indicates that the key sequence has changed in the Windows version.

% **= I**ndicates that the key sequence is new and not available in the DOS version.

1.6.1.1.6 Command Line Commands

The following keystrokes manipulate the Command Line :

Command Line

Key Stroke	**Description**	**Status**
Esc -	Enter command-line mode	
Ctrl-G -	Re-execute command line (Go)	
Ctrl-V -	Copy command line into text	
Ctrl-] -	Change search case sensitivity	
Ctrl-_ -	Reverse search direction	
F1 -	Get last command line	
F2 -	Get next command line	

Ctrl-Brk	Break out of a command-line loop	

1.6.1.1.7 File Command

The following keystrokes handle file I/O:

File Commands

Key Stroke	Description	Status
Ctrl-P	Save and Reload current file.	
Ctrl-PgDn	Move buffer ahead a line	
Ctrl-End	Move buffer ahead a page	

1.6.1.1.8 Mode Commands

The following keystrokes change the mode the editor is operating in:

Mode Commands

Key Stroke	Description	Status
Ins	Toggle insert or overwrite mode	
F3 -	Turn word-wrap mode on/off	
Alt-F3 -	Show status lines (off/on)	
Alt-F7 -	Scroll text (on/off)	
Ctrl-T	Display control chars (on/off)	
Ctrl-Shft-D	Toggle box-character mode	

1.6.1.1.9 Search Commands

The following keystrokes perform search operations:

Search Commands

Key Stroke	Description	Status
Ctrl-F9 -	Word search backward	
Ctrl-F10 -	Word search ahead	
F9 -	Mark location in text	
F10 -	Move to last marker	

1.6.1.1.10 Display Commands

The following keystrokes enable various types of displays in the editor:

Display Commands

Key Stroke	Description	Status
F7 -	Display line number	
Alt-F1 -	Display help summary	
Alt-F2 -	Display ASCII chart	

1.6.2 Deferred Mode

This mode of operation is referred by several different names such as the Command Mode, Deferred Mode or the Command Line. Command Mode lets you write small “Macro” programs that edit text. These programs are called macros. Commands can be combined in many ways to perform almost any conceivable editing task. Groups of characters can be searched out, deleted, new characters inserted, and the operation can be repeated throughout the text. As stated before, the <Esc> character is used to move from Immediate Mode to Command Mode. It is also used to separate certain commands on the command line. Typing two <Esc>’s at the end of the command line executes it. This is a little confusing, so we’ll go through it step by step.

Starting in Immediate Mode, with no commands on the command line, typing <Esc> puts you into Command Mode. You can also click directly on the command line. When you do this, a cursor and a special character “^[“ appear on the command line indicating that you are in Command Mode. Typing another <Esc> executes the empty command line and puts you back into Immediate Mode. Typing a third <Esc> takes you back into Command Mode. As you see, <Esc> takes you back and forth between Immediate Mode and Command Mode.

Here is a list of the topics concerning Command Mode:

The Command Line
Editing The Command Line
Deferred Command Summary
Repeating Commands
Macro Debugger
Searches
Command Loops

Expressions and Conditionals
The Expression Variable
Conditionals
Output Commands
Loops
Aborting Macros/Loops
Saving Macros

1.6.2.1 Deferred Command Summary

Ctrl- = Ctrl key   Alt- = Alt key   Fn = Function key

Delete Commands

Character	**Action**
D	Back a character
R	Rest of line, including CR
Z	Zap back to last form feed
K	Kill entire edit buffer
E	Erase scoop

Insert

Character	**Action**
I	Insert string
X	Insert Scoop Buffer Contents
H	Insert character by its Hex value

File Commands

Character **Action**

G	Get next buffer from disk
A	Append from input file
W	Write to output file
O	Open input and output files
	Close output file
C	

Quit Commands

Character	Action
T	Terminate and discard text
Q	Quit and save text

Miscellany Commands

Character	Action
L	List edit buffer on printer
M	Margin (left, right) or (bell)
Y	Center line
V	Copy text into command line
J	Re-execute command line (Jump)
P	Execute any command line (0-9)
Esc	Command separator
Esc Esc	Execute command line
[]	Enclose a loop
#	Repeat command infinitely
;	Exit a loop

Expression Commands

Character	Action
()	Enclose an expression
[]	Enclose a conditional
$	Hex constant
^	ASCII constant
%	Variable
@	Character after cursor
_	Character before cursor
K	Keystroke
Z	Remaining space
.	Cursor location in text
C	Cursor column position
+ - * /	Add, Subtract, Multiply, Divide
\	Remainder
& !	And, or
~	Not
= #	Equal, not equal
> <	Greater than, less than
>= <=	Greater than or equal, Less than or equal

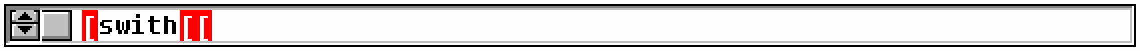
Variable Commands

Character	Action
BA	Insert as ASCII character
BH	Insert as Hex number
BD	Insert as Decimal number
BN	Show on Status Line in decimal
BS	Show on Status Line in hex
BT	Set to scooped decimal
BX	Set to scooped hex
Bn	Save Global (n= 0-9)
-Bn	Load Global (n= 0-9)
:	Move to location in variable
_	Set loop count from variable

Search Commands

Character	Action
S	For a string
Ctrl-W	Wild character
Ctrl-N	Matches any numeric character.
N	Entire file (N edit buffers)
F	Find and replace
+-	Search forward or backward

1.6.2.2 The Command Line



```
[swith[[
```

Command Line

The Command Line is single line edit window at the bottom of the screen. You can type program commands into this editor that can perform complex editing tasks.

You can activate the Command Line for entering text by clicking on it with the left mouse button. If you are typing text into the Edit Screen, you enter the Command Line by hitting Escape.

The Command Line is a miniature editor, which allows you to enter and edit text. For further information about using the Command Line Editor, click here.

Once you have entered one or more program commands, you can execute the commands by typing two Escape characters. After the commands are executed, the editor returns to Immediate Mode and the most recent command remains in place. You can re-execute the command line by clicking on the "Execute" button to the left of the command line. If you are typing text into the Edit Screen, you can also re-execute the command line by typing Control-G.

You can also leave Command Mode, back to Immediate Mode, at any time by typing a Ctrl+X, since this deletes the command line. You also can leave command mode by using the Backspace key to delete all of the characters on the command line.

The Command Line actually has 32 different Command Lines available. Each time you enter a new command line, the last old one is saved for later use. You can cycles through all the different Command Lines, by clicking on the Up and Down arrows on the left edge of the Command Line. You can also cycle through them using the F1 and F2 keys.

To facilitate the use of multiple command lines, XED automatically cycles between them. When command-line cycling is on, hitting <Esc> causes the command-line number to increment before the new command line is entered. This prevents the old command line from being overwritten. This also forms the command lines into a circular buffer. Many times, you want to save the last command line. This way, you have a record of the last ten command lines you have executed.

Any command that works in Immediate Mode also works in Command Mode. For example, Ctrl+A moves the cursor one character forward in both immediate and Command Mode. If you use Ctrl+^ to insert Ctrl+X into the command line, it works like Ctrl+X in Immediate Mode and deletes a line in the edit buffer.

1.6.2.3 Editing Command Lines

The Command Line is actually a miniature editor and you can edit the text there using the mouse and normal editing keys like Backspace, Delete, Insert and the Arrow Keys. Typing Control-^ allows you to insert any character including Control-X.

For more complicated editing of the command line information, you can move the command line information in and out of the Command Line. This is useful because you edit and create complicated editing macros using other edit macros. The most common way of moving commands in and out of the Command Line is using the Windows Clipboard. The editor can Cut, Copy and Paste to and from the Command Line. This is accomplished by highlighting the appropriate text on the Command Line. You can use the mouse or holding down the Shift key and moving the cursor. Once this is accomplished, you can use the standard Ctrl-X, Ctrl-C, and Ctrl-V to do the standard Windows “Cut, Copy and Paste” operations. You can also right click on the Command Line to bring up a menu that allows you to Cut, Copy or Paste.

The other method involves keyboard command that allow you to copy text between the command line and the edit screen. The first command, “V”, is itself executed from the command line. It copies text from the edit buffer into the command line. This way, you can create a macro in the edit buffer then copy it into the command line for execution. The “V” command copies text starting at the cursor. It copies forward from the cursor to the first double <Esc>. (You can insert <Esc> by typing Ctrl+^ followed by <Esc>.)

XED also lets you move macros that are already in the command line into the edit buffer by typing Ctrl+Shift+V. This can be used to modify or repair a macro that’s on a command line.

1.6.2.4 Repeating Commands

All the commands available in Command Mode can be preceded by a number. The number causes the command to be executed that many times. For example, “5D” deletes five characters back from the cursor. If there is no number, or if it is zero, the effect is the same as 1.

Sometimes you want to execute a command over and over until the command fails. For example, you might want to search for and replace some string throughout the edit buffer. You can use a very large number, but the symbol ”#” indicates that a command is to be executed repeatedly until it fails. For example, to find the final occurrence of the word “frog”, type:

```
<Esc>#Sfrog<Esc><Esc>
```

1.6.2.5 Macro Debugger

Macro commands can be difficult to write because they are cryptic and it is easy to be confused about what is going to happen when you execute them. For this reason, there is a Macro Debugger available.

To enable the Macro Debugger, check the “Macro Debugger” checkbox on the Tool Bar. When the Macro Debugger is enabled, the Debugger Dialog Box will appear anytime that the Command Line is executed.

Single Stepping. The main tool that the Debugger gives you is the ability to single-step the execution of a Macro. A copy of the currently executing Macro will appear at the top of the Debugger with a highlight on the Macro command that is about to be executed. Pressing the “Single Step” button causes the program to execute one Macro command and stop. This allows you see what the effect of the current command on the text. If there is a problem with the Macro it will be immediately obvious and you can correct it.

Status Display. The Debugger displays status information about the state of the Macro that is being executed. Here is a description of each item.

1. Decoded Command. Since Macros are single-letter commands that can be cryptic. This item displays a written description of the Macro under the cursor.

2. Status. This item indicates whether the Macro is still running or whether it has completed.

3. Nesting Depth. XED Macros can be nested by calling other Command Lines or by executing Loops. This item displays how deeply nested the Macro is.

4. Char Index. This is the Index of the currently active and highlighted command.

5. Global Variable. This is the numerical value of the “Global Variable.” The Global Variable can is an integer and can be interpreted as Boolean value where 0 is false and any non-zero value is true.

6. Loop Count. All XED Loop Macros can be preceded with count that causes the Loop to be executed a certain number of time. This item displays the Loop Count as it decrements. Note, because of the recursive nature of the XED Loop, the proper count will not be displayed except when the cursor is at the beginning or end of the loop.

7. Character Before/After Cursor. These items display the character that is immediately before and after the cursor. This is useful when you are using the @ and _ character in expressions to test the value of the characters near the cursor.

E**xit and Run Button.** Pressing this button causes the Debugger to close and the Macro to be executed at full speed.

Re-execute Button. Pressing this button resets the Debugger and causes the program to begin single stepping through the Macro again.

Exit and Abort Button. Pressing this button closes the Debugger and aborts the Macro.

1.6.2.6 Searches

XED has several search commands that let you quickly scan for strings of characters. The basic search command looks like this:

```
<Esc>Sanimals<Esc><Esc>
```

If you enter this on the command line, it searches forward from the cursor to the first occurrence of “animals”. If you type Ctrl+G, it is re-executed and the next occurrence of “animals” is located.

You can do several types of searches. For example, searches normally go forward from the cursor, but you can set the search to go backward by typing Ctrl+- (Ctrl+dash). Typing Ctrl+- a second time, changes back to forward searches. The search direction is indicated on the status line by an arrow. The arrow points to the right if the search is forward, and points to the left if the search is backward.

You also can force a search to be forward or backward by placing a “+” or “-“ in front of it. For example:

```
<Esc>-Sfrog<Esc><Esc>
```

This forces a backward search for the string “frog”. This feature is useful with macros when you don’t know the direction that was last set by the Ctrl+- command.

XED lets you change the case sensitivity of searches. Normally, searches are case sensitive, which means that uppercase and lowercase letters are treated as different characters—uppercase “A” does not match lowercase “a”. Typing Ctrl+] turns off case sensitivity so, for example, searching for “COWBOY” also locates the string ”Cowboy”. The case sensitivity is shown on the status line. If “S#s” is shown, searches are case sensitive, and if “S=s” is shown, they are not.

Wild Cards. Sometimes you want to search for a string, but the match doesn’t need to be exact. For example, you might want to locate a string like “the high was 78 degrees”, but the temperature can be any two-digit number. The “wildcard” character, Ctrl+W, is used to do this. When Ctrl+W is placed in a search string, it matches any character. As a result, the search string: “the high was <Ctrl-W><Ctrl-W> degrees” finds the string no matter what temperature was used.

Numeric Wild Cards. You also may want to search for a number, but you don’t care about what the exact value is. For this reason, there is a numeric wild card Ctrl+N. Ctrl+N will match any numeric digit. For example, <Ctrl+N><Ctrl+N> will match any two-digit number.

Often times you want to search for one string and replace it with another. The “F” command is like a search except that it replaces the string with a new string. The syntax of the command is:

```
F <string1> <Esc> <string2> <Esc>
```

XED searches for <string1> and, if found, replaces it with <string2>. The two strings do not need to be the same length. If the second string is empty, XED simply deletes the first string.

Examples. Here are some sample command lines.

```
<Esc>SPRINT<Esc>I LINE<Esc><Esc>
```

This searches forward for the word “PRINT”, and inserts a space and the word “LINE” immediately after it.

```
<Esc>Ftheir<Esc>they’re<Esc>J<Esc><Esc>
```

This command line changes every occurrence of the word “their” to “they’re”.

```
<Esc>S;<Esc>DRI<Enter><Esc>J<Esc><Esc>
```

This macro can be used to delete all comments from an assembly language program. It searches for each occurrence of a “;” character, deletes it and the rest of the line, then inserts a new carriage return to replace the one deleted by the “R” command. This is repeated until the end of the edit buffer is reached.

1.6.2.7 Command Loops

Loops repeatedly execute a group of commands on the command line. They are similar to a “for” loop in a programming. Loops are created by enclosing a group of commands inside a set of braces “ “. If a number is placed in front of the braces, every command inside the braces is executed that number of times. If an error is detected within the loop, the loop aborts immediately without executing any more commands in the braces. For example, if you are doing a search and the search fails the loop stops before executing the rest of the commands inside the loop. Here is an example:

```
<Esc>10SJULY <Ctrl+W><Ctrl+W>, <Esc>I1994<Esc><Esc><Esc>
```

This command line searches for any date in July and adds “1994” to it. It executes 10 times or until it fails to find the search string.

Loops can also be nested. For example:

```
<Esc>2010Sfrog<Esc>Sdog<Esc>Ipig<Esc><Esc><Esc>
```

This command does 10 searches for a complex string then inserts the string “pig”. It does this whole operation 20 times.

If you use a “J” command within a loop the “J” causes the execution to jump to the beginning of the loop, not to the beginning of the whole command line.

Errors cause the current loop to abort. If a loop is nested or if there are other commands on the command line after the loop, XED continues to execute these commands. When there is an error within a loop, XED clears the error as soon as it exits from the loop so it can continue executing commands outside the loop. This is useful for things like searching for the last occurrence of a string. For example, this locates and marks the innermost “begin-end” pair in a block of code:

```
<Esc>#Sbegin<Esc>I *<Esc>Send<Esc>I *<Esc><Esc>
```

Assuming you have a single procedure in the edit buffer, this command does an infinite search for “begin”. When the search fails, the cursor is located just after the last “begin”, which is the innermost “begin”. The code then places an asterisk after the “begin” then searches for the next “end”. This “end” is the corresponding innermost “end” and an asterisk is placed after it.

1.6.2.8 Expressions and Conditionals

XED can do complex editing tasks using expressions and conditional operations. In other words, you can create small programs that do very complex editing tasks. The syntax is relatively simple.

Expressions. Expressions are a series of mathematical or Boolean operations that, when evaluated, result in a single numeric value. XED uses expressions to test for special situations, like a unique combination of characters. Expressions also can be used to create such things as an incrementing column of numbers.

Expressions are always enclosed in parentheses “()”. Expressions consist of two kinds of items: factors and operators. Factors are something that has a value, such as the number 2. Operators operate on factors, like the plus sign operates on 2 + 2.

For a complete list of Factors and Operators, click here.

Note that all these factors and operators can only be used in an expression. In other words, they can only be used inside parentheses. Some of these characters have different meanings outside expressions. For example, “C” means “return the column number of the cursor” inside an expression and “close the file” outside an expression.

When XED evaluates an expression, it reduces all the operators and factors to a single numeric value. For instance, the expression: (1+2+3) reduces to the value 6. Sometimes an expression is used as a Boolean value. In other words, sometimes the expression is reduced to a single true or false value. For example:

```
(@=^A)
```

In this expression, if the character under the cursor is “A”, then the expression evaluates to “true”. In XED, “true” is the value $FFFF and “false” is 0.

1.6.2.9 The Expression Variable

The result of the evaluation of an expression is stored into a special place called the “Variable”. There is only one variable and every time an expression is evaluated, the old value is overwritten by the new value.

You also can set the Variable with a number that is in the scoop. The number is converted from its ASCII representation. For example, the ASCII string “193” is converted to the numeric value 193. There are two commands that perform this conversion:

BX – Interprets text as hexadecimal BT – Interprets text as decimal

These commands search from the beginning of the scoop for a decimal or hexadecimal string (which must be within the first 80 characters or a zero will be returned). The resulting converted number is stored in the Variable.

1.6.2.10 Conditionals

Conditionals are used to selectively perform an editing operation. Conditional operations are indicated using square brackets “[]”. Commands enclosed in square brackets are only executed if the value of the Variable is true. For example:

```
<Esc><Ctrl+A>(@=$1B)[<Ctrl+A>D]<Esc><Esc>
```

This command line deletes escape characters ($1B), something that normally cannot be done. The command line starts by moving the cursor forward one character. If the cursor is on an escape character, the commands in the square brackets are executed. These commands move the cursor forward one character then delete the escape.

Conditionals can be nested, or placed inside loops. They cannot have a numeric value in front of them like other commands. For example, this would give an error:

```
<Esc>234[Ifrog<Esc>]<Esc><Esc>
```

1.6.2.11 Output Commands

XED has several commands used to display or insert values:

BA – Inserts the Variable as an ASCII character
BD – Inserts the Variable as a decimal number

BH – Inserts the Variable as a hex number
BS – Shows the value of the Variable on the status line in hex
BN – Shows the value of the Variable on the status line in decimal

Three of these commands insert values into the edit buffer. As you will see in the examples below (10.2.7 Examples), this is useful for creating a series of numbers in your text. The “BS” and “BN” commands are useful for debugging command strings. They also can be used to display the ASCII value of any character in your text.

1.6.2.12 Loops

If you combine expressions and conditionals with loops, you can create some powerful macros. To facilitate the use of conditionals with loops, XED has the semicolon “;” command. This forces a loop to terminate at the point it is encountered. This provides a way to terminate a loop based on a condition. For example:

```
<Esc>#<Ctrl+A>(@=$1B)[<Ctrl+A>D;]<Esc><Esc>
```

This command searches through the text for an escape character ($1B). When it finds an escape, it deletes it and stops.

The command iteration count is normally set by putting a number in front of a command. For example:

```
<Esc>10Inow is the time<Esc><Esc>
```

This inserts ten copies of “now is the time” into the buffer. However, you also can set the loop iteration count using the Variable. This enables the number of iterations to be calculated with an expression or to be set by a condition. The underline character “_” copies the value of the Variable into the iteration count. Since the value of the Variable is set by the value of the last expression evaluated, any command can be executed a calculated number of times. For example, the following command can be used to pad a string with spaces:

```
<Esc>S<Enter><Esc><Ctrl+Q>(8-C)<Ctrl+W>_I <Esc><Esc>
```

This command right-justifies a string that could be up to eight characters long. The string is assumed to be on a line by itself. The first command searches for the carriage return that terminates the line. The next command backs up to the last character of the string. The column position is then subtracted from 8. This gives the number of spaces needed to justify the string. The cursor is then moved to the beginning of the line, and the proper number of spaces are inserted.

Cursor Movement. Another command moves the cursor to a location in the edit buffer depending on the value of an expression. The colon command “:” moves the cursor to the location specified by the Variable. For example:

```
<Esc>(@=<^a>)[(.+20):]<Esc><Esc>
```

This command line moves the cursor 20 locations ahead if the character under the cursor is “a”.

9.2.7 Examples

The following examples illustrate the power of expressions and conditionals.

Display the value of the character under the cursor in hex:

```
<Esc>(@)BS<Esc><Esc>
```

Search for and delete escape characters:

```
<Esc>#(@=$1B)[<Ctrl+A>D;]<Ctrl+A><Esc><Esc>
```

Search for the first occurrence of a numeric character:

```
<Esc>#<Ctrl+A>(@>=$30 & @<=$39)[;]<Esc><Esc>
```

Search for the word “the” and if the keystroke is “+” then delete it.

```
<Esc>Sthe<Esc>(K=<^+>)[DDD]<Esc><Esc>
```

List the numbers 0 through 99 on the left side of the screen:

```
<Esc>(0)100bdi<Enter><Esc>(%+1)<Esc><Esc>
```

Count the number of "a" characters in the text and display the number on the status line:

```
<Esc>(0)#sa<Esc>(%+1)bn<Esc><Esc>
```

Move to the 12th column on the first line with more than 11 columns:

```
<Esc><Ctrl+S>#<Ctrl+A>(C>=12)[;]<Esc><Esc>
```

List the numbers 0 through 255 in hex on the left side of the screen:

```
<Esc>(0)255BHI<Enter><Esc>(%+1)<Esc><Esc>
```

Display the hex code and the ASCII character for the full character set:

```
<Esc>(0)256BHI - <Esc>BAI<Enter><Esc>(%+1)<Esc><Esc>
```

1.6.2.13 Saving Macros

Last Command Line. XED automatically saves the last command line when you close the editor. The next time you run the editor, the last command line is restored. This is very handy if you are searching several files for the same phrase or performing the same operation on several files.

Multiple Macros. In addition, XED allows you selectively to save up to 30 command line macros to a file. Once they are saved, they can be selectively restored and put to use in a variety of tasks. You can also move, delete and generally re-organize the macros to meet your needs.

To work with macros, select the "Tools|Macros" option from the menu bar. This will display the "Macro Tools" box.

Working With Macros. The central part of Macro Dialog Box consists of the Macro Grid. Macros that you create in the normal course of editing can be moved to the Macro Grid. You can then name and organize the macros and when you are ready, save them to disk. Once the macro have been save to disk, they can be restored and used anytime you want.

Here is a detailed explanation of each section of the Dialog Box:

1. Macro Grid. The Macro Grid displays a complete list of all the available Macros. Each line in the grid displays one macro. Each Macro can have a name of up to 20 characters. This helps you identify the purpose of the macro. To enter a macro name, you simply click on the cell and type the name. You can also enter the cell by typing F2. Macros cannot be entered directly into the grid. They must created on the Command Line and copy to the Grid.

2. Load Button. This button allows you to load a previously saved macro files. Macros are normally saved to the "Default Macro Folder." The Load Dialog automatically go to this folder first. You can set up the "Default Macro Folder" using the Options Dialog Box.

3. Save Button. The Save Button allows you to save the current list to the current file. Normally the files have the extension "XED" to distinguish them from other files. Macros are normally saved to the "Default Macro Folder." The Load Dialog automatically go to this folder first. You can set up the "Default Macro Folder" using the Options Dialog Box.

4. Save As Button. The Save Button allows you to save the current list to a specific file. Normally the files have the extension "XED" to distinguish them from other files. Macros are normally saved to the "Default Macro Folder." The Load Dialog automatically go to this folder first. You can set up the "Default Macro Folder" using the Options Dialog Box.

5. Up And Down Buttons. These buttons move the currently selected macro up and down in the list. This is useful for move the most commonly used macros to the top of the list and general reorganizing.

6. Delete Button. This button deletes the currently selected macro.

7. Get Command Line Button. This button moves the current Command Line into the currently selected Macro Grid row. This is the main method of adding new macros to the palette of saved macros.

8. Set Command Line Button. This button moves the currently selected macro into the current Command Line. This is the way you utilize macros. Once they are moved to the Command Line, they can be executed using the] usual Control-G and mouse click commands.

9. Get All Commands Button. This button moves all the macros in all the Command Lines into the Macro Grid. The macros are inserted at the currently selected row. Macros already in the grid are moved down the list making room from the new macros. As a result, some macros may be pushed off the end of the list and lost.

10. Set All Commands Button. This button copies 10 macros out of the Macro Grid into the set of Command Lines. The 10 macros are taken from the Grid starting at the selected row.

11. Change Command Line Button. These two buttons increment and decrement the current Command Line. This is useful for selecting the proper Command Line to transfer to and from the Macro Grid.

1.6.2.14 Abort Command

It is possible to write macro commands that are infinite loops. You can also write commands that, although they are not infinite, take hours or days to complete. XED lets you exit from these loops by typing Ctrl+Break. When you hit Ctrl+Break to abort a loop, the message "break" appears on the status line.

1.6.3 Unimplemented Commands

Several commands that exist in the DOS version remain unimplemented in the Windows version. Some are irrelevant to the Windows version. Some are archaic and seldom used and some may be implemented at a later date.

Search Commands

Character	Action
/	List help summary on printer
U	Go to DOS
Alt-Q	Confirm this command line
Alt-F8	43/50 Line mode (on/off)

Obsolete File Commands

Character	Action
A	Append. With virtual buffers, the original function is no longer relevant. As result, in this version, the "A" command is used append text from other files.
G	Get Next Buffer. With virtual buffer no longer meaningful and has been dropped.
N	Cross Buffer Search. Since there is only one buffer, this command is obsolete and you should call the "S" command
W	Write Buffer. Obsolete and superseded by the "Save" and "Save As" commands.
Ctrl-Page Down/F4	Shuffle Text in and out of buffer. Obsolete and eliminated.

Command Line Switches

Character	Action
/L	Line feed stripping
/N	Non-scrolling mode
/F	Full screen mode
/P	Protect mode
/Q	Quiet the bell
/C	XED command line. (Use ` in place of ESC key. ^ for carriage return.)
/?	Show these switches

1.6.4 Command Line Key Commands

The Command Line works like a simple editor, which allows you to edit command text. There are several special characters and commands that work on the Command Line. Here is a detailed description of each character and command:

1. Escape. The Escape <Esc> character is used extensively on the command line. If can be used to separate some commands and it is also used to execute the commands.

2. Left Arrow/ Right Arrow. The Left and Right Arrow keys move the cursor to the left or right.

3. Backspace. This key deletes the character immediately to the left of the cursor.

4. Insert. This key toggles the Command Line Editor between Overwrite Mode and Insert Mode. In

Overwrite Mode, characters typed into the keyboard delete old character under the cursor. In Insert Mode, characters are shifted to the right as new characters are inserted.

5. Delete. The Delete Key deletes the character immediately to the right of the cursor.

6. Ctrl-O. This command “undoes” any recent changes.

7. Ctrl-Shift-O. This command “redoes” any recently “undone” commands.

8. Ctrl-X. If text is highlighted, the command deletes and copies the text to clipboard. If no text is highlighted, the command deletes from the cursor to the beginning of the command line and returns the focus to the Main Edit Screen.

9. Ctrl-C. This command copies any highlighted characters to clipboard.

10. Ctrl-V. This command pastes or inserts the contents of the clipboard at the cursor.

11. Shift Key. Holding down the Shift Key and moving the cursor causes the intervening text to be highlighted.

1.7 Advanced Operations

XED has many features that allow you to carry out complex tasks on a document. Here are detailed descriptions of these tasks:

Find and Replace
XED Fonts
Defaults
Pair Matching
ASCII Character Chart
Screen Colors and Fonts
XED Options and Settings
Setting Case
Block Indent
Margin Presets
Reopening Files
Popup Menus

Importing Files
Replacing Tabs/Spaces
Sorting Text
Removing HTML
Insert/Remove Line Marks
Setting Spanish Characters
Spell Checking
Executing External Programs
Unicode
Converting 8-bit ASCII to 7-bits
File Details
Merging Files
Opening URLs in the Text

1.7.1 Find Replace

XED has two methods to Find and Replace text. The first method uses the standard XED “S” and “F” commands. These are described in the Deferred Command section.

The second method uses standard Windows “Find” and “Replace” dialog boxes. You can access these methods by selecting “Edit|Find” or “Edit|Replace” from the menu bar. The main advantage of these commands over the standard XED is they give you more control over the replacement process. For example, you can selectively replace onlycertain instances of a word with this feature.

To make the Find and Replace dialogs easier to use, XED automatically loads the search and replace strings for you. If you have highlighted a block of text, the Search Strings are taken from the highlighted text. If you have just completed a Command Line Search or Search and Replace, the arguments are taken from the Command Line. If you have just completed a Command Line Insert command, the Replace String will be taken from the Insert String.

1.7.2 XED Fonts

XED allows you to use different fonts in the display and with the printer. To change fonts, select “Preferences|Colors/Fonts” and then press the “Set Fonts” button. This will present the standard Windows Fonts dialog box. The box allows you select from a number of fonts and sizes.

Font Types. Windows supports many different types of fonts. XED can use many of these different font types with a few restrictions. Here are some of things you need to know about fonts:

1. Fixed Pitch Fonts. XED can only use Fixed Pitch fonts. Fixed Pitch fonts are fonts where each character has the same width. This is different from variable pitch fonts where the thin character like “I” takes up less space than a wider character like the “W.”

2. Oem Fonts. OEM fonts are fonts that have the same standard characters set as the VGA, EGA, CGA and MDA video cards in text mode. The extended ASCII character (those with a hex value greater than $7F) contain the “box character” set used for drawing square boxes on the text screen. If you want view or draw box characters, you must use an OEM font. The standard OEM character set for Windows is called “Terminal.”

3. Ansi Fonts. ANSI fonts have foreign language character in the extended set and don’t have any symbols associated with control characters.

4. Other Font Types. Windows also supports several other font types such as “True Type,” “Raster,” and “Vector.” All of these font type can be used with XED as long as they are Fixed Pitch.

1.7.3 Default Settings

XED allows you to save all of your settings to a standard Windows initialization file. Whenever XED runs, it reads the initialization file and sets the editor accordingly. This enables you to customize the program so that it works the way you want it to.

To save the current settings, select “Preferences|Save User Settings” from the menu bar. These setting will be used the next time you run the editor.

You can also Restore your setting anytime you wish by selecting the “Preferences|Restore User Settings” option from the menu bar. This is useful where you have made temporary changes to the settings and wish to revert back to the original.

Finally, you can restore the original “Factory Default” settings by selecting “Preferences|Use Factory Settings” from the menu bar. The factory default setting restores the editor to generic settings such as white screen, a black font, etc.

Currently, XED save 30 items including: screen margins, paragraph indent, bell column, printer white space, automatic line wrapping, the style for the Control-N, Control-B, and Control-C commands, search case sensitivity, scrolling mode, the font color, screen color, highlight color, control character color, and font characteristics. It also saves the Tool Bar, Scroll Bar, Ruler, Command Line, and Status Line visibility. The information is saved in the file “XEDIT.INI” which can be found in the WINDOWS directory.

1.7.4 Pair Matching

XED has the ability find the matching “end” of any kind of “begin-end” pair. This means it will search over nested pairs to find the matching “end” to a specific “begin”. This feature is useful for finding instances of mismatched pairs in program source code and markup languages. For example, XED can find the “begin-end” pairs in Pascal and XPL, the “ – “ pairs of C, C++ and RTF files and pairs like “<b> - <\b>” from HTML files.

To look for the matching “end” symbol of a particular “begin”, position the cursor before the specific “begin.” Then select the “Tools|Match Pairs” option from the menu bar. This will present the “Match Begin-End Pairs” dialog box.

There are two edit boxes that allow you to enter the specific “begin-end” pair strings. Just enter the “begin-end” strings, push the “Find Matching” button and the program will automatically find the matching “end”. If there is no matching the “end”, the program will display an error message.

You can also step through the text, locating “begin” and “end” symbols. The program keeps track of the nesting level and displays the depth. Each time a “begin” symbol is encountered, the depth value is incremented. Each time an “end” symbol is encountered, the depth value is decremented. In this way, you can keep track of the nesting level, looking for places where it becomes unbalanced.

You also have options that control the symbol search. For example, you can control the case sensitivity of the search and whether the search will find whole work matches.

1.7.5 ASCII Chart

You can display the ASCII character chart by typing Alt-F2 or selecting “View|Character Set” from the menu bar. The ASCII Chart displays a grid of all 256 characters available in the current font. This allows you view the symbols used for control characters and the characters in the extended ASCII set. The hexadecimal value of each character is displayed in the grayed rows and columns along the top and sides, with high nibble on the top and the low nibble along the side.

You can select any character in the set my clicking the mouse cursor the appropriate cell. You can then insert this character into the editor text by pressing the “Insert” button.

1.7.6 Screen Colors And Fonts

XED give you complete control over all the screen colors and the font used to display the text. To make changes in these settings, select “Preferences|Colors/Fonts” from the menu bar. This will display a dialog box that allows you to change these options.

Colors. At the top of the dialog box is a list of four screen colors. You can change these colors by clicking on the specific text color item and then selecting the appropriate color from the grid on the right. Here is an explanation of each color item.

1. Text. This is the color of the individual characters in the text.

2. Screen. This is the color of the screen background.

3. Highlight. This is the color of highlighted text when you mark the text by dragging the mouse.

4. Control. This is the color of control characters.

Normally, the colors are set so that there are black characters with a white background, simulating printing on a white piece of paper. One option is to set a black background with light gray text to simulate the DOS version of XED.

Fonts. The fonts used in XED can be set by clicking on the “Set Fonts” button. This will display the normal Windows Font Dialog Box. The Dialog Box will display all available fixed pitch fonts. For more information on Windows Fonts, click here.

Preview. The Preview section shows how the various font and color selections will look on text. Note that when control characters are highlighted that the color will be the exclusive “OR” combination of the highlight color and the control color.

1.7.7 XED Options

XED allows you to control various aspects of the way the editor works. To access these options select “Preferences|Options” from the menu bar. The Options Dialog has three sections. You can select each option by clicking on the appropriate tab at the bottom of the box.

Display Options. There are three options that control the way the text displayed or manipulate:

1. Show Cursor Line. When this option is enabled, the line where the cursor located is highlighted by with a light colored background. This makes it easy to find the working line and make it easy to line up text on the right and left sides of the screen. The highlight color can be set using the Colors/Fonts Option.

2. Linear Horizontal Scroll. When the text cursor reaches the right edge of the screen, the screen scrolls to keep the cursor in view. Normally, the screen will not scroll back until the cursor reach the left edge of the screen. If Linear Scroll is enabled, the screen scrolls back as soon as the cursor moves away from the right edge.

3. Cursor Width. The width of the flashing cursor can be set to any value from one to 30 pixels. You do this by click on the up and down-arrow next to the box or by typing in the a value. Narrow cursors give a sense of control, where a wider cursor is more visible. You can also the cursor width to match the font you are using by clicking on the Character Width button.

4. Default Screen Size. The starting size of the XED screen can be configured to match your tastes. Clicking on the “Use Current Size” button will cause the current size of the Window to be saved as the default. The next time the editor is run it will start at this size. The screen size is not saved using the Save User Settings menu command. This prevents the default screen size being set by randomly when other defaults are save.

5. Reopen List Size. The maximum number of files in the Reopen Menu list defaults to eight. This option allows you to increase or decrease this number. Remember, in order for this value to become the default, you must select “Options|Save User Defaults” from the menu bar.

6. Spaces Per Tab. This option controls the number of number of spaces used to display a tab. The value can be anything between 1 and 20 spaces. The value is used for display, printing, converting tabs to space and converting spaces to tabs. The value can be saved and made the default using the Save User Settings menu command.

Tools. The Tools section controls the display of various visual tools that appear around the edit screen. These options allow you to display or hide each tool. Here is a complete description of each:

1. Tool Bar. The Tool Bar appears along the top of the screen and it holds various shortcut buttons that perform editing actions.

2. Ruler. The Text Ruler appears along the bottom of the screen and indicates the current column position of the text and the margins. For more information on the Ruler click here.

3. Command Line. The Command Line appears along the bottom of the screen and allows you to enter programming commands. For more information on the Command Line click here.

4. Status Line. The Status Line appears along bottom of the screen and gives you status information about the current state of the editor. For more information on the Status Line click here.

5. Scroll Bars. There are two scroll bars and one pair of scroll buttons around the edge of the screen. You enable each scroll bar separately by clicking on the appropriate check boxes. For more information on the scroll bars, click here.

Actions. This section controls the way editing-action work.

1. Word Move/Delete Style. XED has several keys-stroke commands that move over or delete whole words. There have been two styles of doing this. The first style moves or deletes to the beginning of the word. The second style moves or deletes to the beginning of the separators between words.

2. Highlight Search Results. When this option is enabled, XED highlights the results of each search, making it easier see. Also, since the text is highlighted, you can immediately delete, overwrite or copy to the Windows clipboard.

3. Append Eof Character To Files. This option controls whether XED will append an EOF character (Control-Z) to the end of a file. Some programs and operating system use an EOF character to mark the end of a text file. However, some programs will display the EOF character as box or black bar at the end of the text. This option allows you to work with programs that don’t recognize EOFs.

4. Auto Save. This option allows you to automatically save the contents of the edit buffer at certain time intervals. This assures that you text will always be saved incase there is a power failure or your computer crashes. The program won't attempt to auto save if the text hasn't changed or there isn't an output file. You can enable or disable the option. You can also set the interval between automatic saves from between 1 and 1440 minutes (24-hours). You can make the setting the default by selections the "Preferences -> Save User Settings" option from the menu bar.

Directories. This option allows you to setup default directories for XED. Here is list of the Directories and their functions:

1.Default Directory. This is the default directory when the program first starts running. If you create a new file and then try to save it, the Save Dialog will show this directory first. After you change this directory, you must select the "Options -> Save User Settings" item from the menu bar to make the change permanent.

2. Macro Directory. This is the default directory to save XED Macro Files.This directory will be the first directory shown when you load or save a macro.

1.7.7.1 Scroll Bars

XED has two scroll bars on the bottom and right hand side of the edit screen. The scroll bars are used to scroll the text both horizontally and vertically. The each scrollbar can be individually hidden to give you more open editing area using the Options dialog box.

Early versions of XED for Windows could not scroll the text without moving the cursor or turning off any highlights that might be on the screen. The current version works like most GUI editors, in that the scrollbars only scroll the text, they do not move the cursor or effect the highlights.

1.7.8 Setting Case

XED allows you to change the case of a block text all at once. To do this, highlightthe block of you wish to change and select “Format|Change Case” from the menu bar. The will display the “Set Case” dialog box. By click the “Apply” button, you convert the case of the highlighted text.

There are five options for changing case:

1. Upper Case. This option switches all highlighted text to upper case.

2. Lower Case. This option switches all highlighted text to lower case.

3. Sentence Case. This option switches the first character of each sentence to upper case. The rest of the text is switched to lower case.

4. Title Case. This option switches the first character of each word to upper case and the remaining text to lower.

5. Toggle Case. This option toggles all characters in to the opposite case.

1.7.9 Block Indent and Unindent

XED allows you to indent or un-indent a block text all at once. To do this, highlightthe block you want to change and select “Format|Block” from the menu bar. The will display the “Block Indent” dialog box.

The dialog box gives you the choice of using either a tab or space to indent with and the number of characters to use. It also gives you the choice of Indenting or Un-indenting. If you choose Indenting, the program will add the specified number of spaces or tabs in front of each highlighted line. If you choose Un-indenting, the program will remove the specified number of tabs or spaces from the beginning of each line. When un-indenting, the program will only remove the specified space or tab characters from the beginning of each line.

1.7.10 Margin Presets

XED has the ability to retain four Margin Presets. You access these presets by Right Mouse Button clicking on the Text Ruler. When you do this, a menu will appear showing the four available margin settings. To change the margin settings, just select one of the settings. This allows you to quickly switch between any of the four different margin settings.

You can also edit the settings by selecting the “Preference|Margins” option from the menu bar. This will bring up the Margins dialog box, which shows a list of each of the four margin settings. To change the settings, just enter the new values in the appropriate cell. You can also use one of the settings by highlighting one of the rows and clicking on the “Apply” button. In addition, clicking on the “Insert Current Margin” button will cause the current margin settings to be stored in the highlighted row.

Finally, the current margin settings can be made the default setting by selecting “Preferences|Save User Settings” from the menu bar. Likewise, the factory default margins and the users margins can be restored using the “Preference|Restore User Settings” and the “Preferences|Use Factory Settings” menu options.

1.7.11 Reopen Menu

The Reopen menu keeps track of the last few files that you have edited in XED. A list of these files is displayed in the “File|Reopen” menu item. You can quickly reopen any of these files by simply clicking on them. You can control the maximum number of items in the list from the Options Dialog Box.

1.7.12 Popup Menus

XED has several menus that “Popup” when right mouse button click on an area of the screen. If you right mouse button click on the main XED Screen a menu will appear that allows you to Cut, Copy and Paste between XED and the Windows Clipboard. Likewise if click on the Text Ruler a menu will appear that allows you to select one of four preset margins. Finally, if click on the Command Line a Cut, Copy and Paste menu will also appear.

1.7.13 Importing Files

Different operating systems and computers use different formats for text files. For example, DOS and Windows use both Carriage Return and Line Feed to indicate the end of a line whereas Unix/Linux uses a Line Feed alone. XED can correctly read either format. Currently, XED saves files with both a Carriage Return and Line Feed as the line terminator.

1.7.14 Replacing Tabs or Spaces

It is often useful to replace tabs in a document with spaces. For example, some programs may not display tabs correctly so that the columns align. Also, it can be useful to replace spaces with tabs. For example, assembly language programs are easier to edit and small if tabs are used.

XED allows you to replace tabs with spaces or spaces with tabs, while maintaining the original column alignment. To replace tabs with spaces, highlight a blockof text and then select the “Format|Tabs To Spaces” option from the menu bar. To replace spaces with tabs, highlight a block of text and then select the “Format|Spaces To Tabs” option from the menu bar.

1.7.15 Sorting Text

It is often useful to sort text into alphabetical order. XED allows you to do this by highlighting the text and then selecting “Tools|Sort Highlighted.” The highlighted text will be sorted into alphabetical order.

Note: because the routines sort alphabetically, left-justified numbers may not be sorted correctly. This caused by the fact that an alphabetical sort will compare leading digits, even though they may represent ones, tens, hundreds or thousands. For example, 20 will be placed before 7 because the 2 is less than 7. You can solve the problem by inserting leading zeros or leading spaces.

1.7.16 Removing HTML from Text

If you are working with or downloading HTML documents, it is often useful to be able to remove the HTML commands. This enables you to edit or display the text outside of a web browser.

To do this, highlight the block of text you are working with and select the “Format|Remove HTML” option from the menu bar. This will remove all of the HTML commands and leaving behind the raw, displayable text. Note: HTML commands that are not associated angle-bracket pairs will not be removed. For example, command like “&nbsp;” will not be removed.

1.7.17 Insert/Delete Line Marks

Email programs often add marks at the beginning of each to indicate that the text is a part of a previously message. XED allows you to add or removes these marks. To add marks, simply highlight the text you want marked and select the “Format|Insert Line Marks” from the menu bar. Likewise, if you want to remove Line Marks, simply highlight the text and select the “Format|Remove Line Marks” option from the menu bar.

1.7.18 Setting Spanish Characters

You can generate Spanish text markers from English text. You do this by highlighting a block of text and then selecting “Format|Convert To Spanish” from the menu bar. Any time XEDIT encounters a tilde (~) or apostrophe (‘) it deletes the character and converts the character before to the appropriate Spanish characters.

1.7.19 Open URLs in the Text

You can highlight any URL in the text and open it in your browser.You do this by highlighting a URL and then selecting the "View -> Open Highlighted as Web Page" option from the menu bar. You can also get to the same menu options by right clicking on the text.

1.7.20 Spell Checking Text

XED has the ability to spell check text. Here is detailed information about spell checker:

Spell Checking
Spelling Options
Custom Dictionaries

1.7.20.1 Checking Spelling

Spellchecking. To spell check your text, you simply select “Spell Check” from the “Tools” menu. Spell checking begins at the cursor and terminates at the end of the text. When a spelling error is found, a dialog box will be displayed showing the error and other information.

Suggestions. The program tries to find words that most closely approximate the misspelled word and the best matches are displayed in the “Suggestion” list to left. The top rated suggestion will be shown in the “Replacement” box.

If you don’t agree with the suggestion, you can select another or you can enter your own correction. You can also find additional suggestions by entering new spellings in “Replacement” box and then pressing the “Suggestions From” button to the left.

Split Word Button. When you press this button, the program will try to find suggestions based on splitting a misspelled word into two pieces. For example, pressing the button will cause the Spell Checker to offer “every time” for “everytime”.

Hyphenated. This option is similar to the “Split Word” option above. When you press this button, the program will try to find suggestions based on splitting the misspelled word and inserting a Hyphen.

Correcting. Pressing the “Replace” button replaces the text with the suggestion or your own correction. Pressing “Ignore” leaves the text unmodified and proceeds with the checking process. Pressing “Ignore All” causes the spell checker to ignore all occurrences of a particular word until the next time the editor is run. Pressing “Cancel” aborts the checking process.

Saving New Words. You have the option of saving new words to the current dictionary. To save a word that was flagged as misspelled, press the “Add Not Found” button. You can also save words that you type into the “Replacement Box, by pressing the “Add Replacement” button.

1.7.20.2 Spelling Options

Options. You can configure the Spell Checker by selecting the “Preferences|Spell Checker” option from the menu bar. Here is a detailed description of all the options:

Ignore Words With Numbers. When this option is enabled, the spell checker does not examine words that contain one or more numerical digits. For example if the option were enabled, the word “K2” would not be tested. This is useful because the majority of words containing numbers are not normal words and would not be found in the dictionary.

Ignore Uppercase Words. When this option is enabled, the spell checker does not examine words that are all upper case. For example, if the option were enabled, the word “YMCA” would not be tested. This is useful where you have a large number of acronyms that are unlikely to be found in the dictionary.

Current Dictionary. This option allows you to select the current dictionary being used by the program. When the program is first installed, it uses a default dictionary called “XEDSmall.dix” which will be found in the same directory as XED itself. If you wish to select another dictionary, you can do this by entering a filename and path in the box or by browsing for the file. You can make the current dictionary the default by using the Save User Defaults option.

The default dictionary comes with about 15,000 words, which misses a lot of common words, but it is much faster to search and generate suggestions, particularly on a slow computer. XED also comes with a larger dictionary called “XEDLarge.dix”, which contains more than 60,000 words. However, it can be annoying to use on a slow computer.

Normalize Dictionary. This option formats the text in the dictionary to match the requirements of the spellchecker. It does this by removing any leading or trailing spaces, removing duplicates and sorting the list of words.

1.7.20.3 Custom Dictionaries

Custom Dictionaries. It is relatively easy to create your own custom dictionaries. The file format for a dictionary is simply one word per line, with all words in lowercase. The words are also expected to be sorted, although the spellchecker will automatically sort the words if they haven’t been presorted. If you have a dictionary where the format is not correct, the spellchecker can convert it to the correct format using the Normalize Command.

1.7.21 User Tools - External Programs

It is sometimes useful to run other programs directly from the XED. This can extend the capabilities of XED and integrate XED with other programs. For example, it can be useful to execute a compiler from inside XED. That way you can edit a program and compile it without leaving XED.

To accomplish these operations, XED has the ability to attach any DOS or Window program to its menu. Commands can be added to insert arguments into the command line for the external program being executed. This makes it easy to perform tasks without searching through folders or fumbling with the task bar.

You can add new tools to the XED's menu by selecting the “Tools|Add User Tools” from the menu bar. When you do this, the “External Programs” window will appear. It consists of two pages:

Edit Page - Attaching and Editing Programs
Debug Page - Debugging Programs.

Executing External Programs. Once a program is attached, a new menu item will appear on the menu bar just below the “User” item. Clicking-on the menu item will execute the program.

1.7.21.1 Attaching Programs

The Edit Page allows you to attach and edit programs. Here is a detailed description of the items on the page.

Program List. On the left side of the window is a list of programs. These are the programs that will appear in the menu. You can add, edit, move or delete program items. Here are detailed descriptions of the options:

Adding Items. You can add new program items by click-on the “Add” button. When you do this, the bottom half of the window will expand to reveal the “Editing” area. To add a new item, you simply enter the “Title” and the filename of the program. The “Title” is the name that will appear in the menu. The filename is the drive, directory and name of the executable file that will be run. To make it easy to find programs, you can browse through your disk using the “Browse” button.

Short Cuts. You can create a “Short Cut Key” for the item by placing an ampersand “&” character in front of one of the letters in the title. The letter following the ampersand will be underlined when it appears in the menu and you can select the item by typing that key. For example, the item “&Database” will be displayed as “Database”, with the “D” acting as a shortcut.

Parameters. Many times, you will need to send parameters to the program you are executing. Usually, this will be the name of the file that is being edited in XED. This allows you to do things like compile the program that is being edited in XED. To do this, you simply insert “%” followed by a letter. These two letters will then be replaced by the corresponding parameter. Here is a list of the parameters:

%w – The file and pathname of the file being edited by
%p – XED.
%f – The path of the file being edited by XED.
%n – The filename of the file being edited by XED.
%e – The filename without the extension.
%o - The extension alone.
The file and pathname of the file without the
extension

Commands. In addition supplying command line parameters, you can also give command to the editor that control the way the external program interact. Here is a list of the commands:

%s – Save current file before running external program
%l – Use return code from the program as a line number

Finishing Up. When you have entered a title and selected a program, click-on the “Done” button, and the new item will appear in the “Program List.” 2. Editing Items. You can edit existing program items by clicking-on the desired item and then clicking-on the “Edit” button. When you do this, the “Editing” area will appear and you will be able to edit the Title and filename. (See “Adding Items” above.) 3. Deleting Items. You can delete any item by selecting it and clicking-on the “Delete” button. 4. Moving Items. You may want to put the most commonly used tools at the top of the menu for easy access. The items will appear on the menu in the same order they appear in the list. For this reason, you can move the selected item in the list by clicking-on the up or down button.

1.7.21.2 Debugging Programs

The Debug Page is designed to help you debug the attached programs to see why they may be failing. The page logs whenever you attempt to use one of the items on the "User" menu and displays the exact commands that used to execute the program. This will tell you exactly what commands are being using to run the program and what arguments are sent to the program on the command.

It will also show the value of the argument returned by the external program when it finishes executing.

1.7.22 Unicode

Unicode is a special type of text file that is used to display unusual characters and non-English alphabets. Ordinary ASCII characters are composed of 8-bit characters, which allows for only 256 characters. Unicode characters can be up to 32-bits in size and can provide billions of characters.

The most common type of Unicode character sets use 16-bit characters, which is more than enough for most purposes. With the 16-bit characters sets, if the high byte of a character is zero, the lower 8-bit part corresponds to the normal ASCII characters set. To identify Unicode files, they have a “Byte Order Mark” (BOM) at the beginning of the file. The BOM indicates whether the file uses Big or Little Endian format and the number of bytes per character.

When XEdit reads a file, it checks to see if the BOM can be found at the beginning of the file. If it can, it marks the file as Unicode and translates the file so it can viewed and edited in XEdit. It also displays the word Unicode on the status line. If you save the file, it will be saved as Unicode. (At this point, the program will only save files as 16-bit big endian files.)

You can convert a file from ASCII to Unicode and vice versa using the File Details dialog options.

1.7.23 Convert 8-bit to 7-bit ASCII

The original ASCII character set used 7-bits and encompassed 128 characters. However, most modern computers are byte oriented which means there are an addition 128 characters available. Early DOS character sets used these characters for box characters to help draw primitive windows on the screen. With the advent of true GUI, Windowing environments, the box characters became less useful and most people switched ANSI character sets that generally used the extra 128 characters for enhanced characters such as non-English characters and special symbols.

It is still sometimes useful to confine things to the original 7-bit ASCII character set. For this reason, XED allows you to convert extended characters into ordinary ASCII characters. To do this, highlight the text and the select the "Format -> Convert 8-bits to 7-Bits" option from the menu bar. The program will then convert all the characters whose values are greater the 127 to an equivalent 7-bit character.

Generally the program tries to convert the character to the nearest equivalent 7-bit character. Here is a chart showing how each symbol is converted. All other characters are converted to spaces:

‘ = ($91) - `
’ = ($92) - "
** = ($A0) -**
¡ = ($A1) - !
¢ = ($A2) - cents
£ = ($A3) - pounds
¤ = ($A4) -

À = ($C0) - A
Á = ($C1) - A
Â = ($C2) - A
Ã = ($C3) - A'
Ä = ($C4) - A
Å = ($C5) - A
Æ = ($C6) - AE

à = ($E0) - a
á = ($E1) - a
â = ($E2) - a
ã = ($E3) - a
ä = ($E4) - a
å = ($E5) - a
æ = ($E6) - ae

¥ = ($A5) - y
¦ = ($A6) - |
§ = ($A7) - s
¨ = ($A8) - -
© = ($A9) - (c)
ª = ($AA) - a
« = ($AB) - <<
¬ = ($AC) - -
** = ($AD) - -**
® = ($AE) - (R)
¯ = ($AF) - _
° = ($B0) - degrees
± = ($B1) - +/-
² = ($B2) - 2
³ = ($B3) - 3
´ = ($B4) - "
µ = ($B5) - u
¶ = ($B6) - P
· = ($B7) - .
¸ = ($B8) - ,
¹ = ($B9) - 1
º = ($BA) - 0
» = ($BB) - >>
¼ = ($BC) - 1/4
½ = ($BD) - 1/2
¾ = ($BE) - 3/4
¿ = ($BF) - ?

Ç = ($C7) - C
È = ($C8) - E
É = ($C9) - E
Ê = ($CA) - E
Ë = ($CB) - E
Ì = ($CC) - I
Í = ($CD) - I
Î = ($CE) - I
Ï = ($CF) - I
Ð = ($D0) - D
Ñ = ($D1) - N
Ò = ($D2) - O
Ó = ($D3) - O
Ô = ($D4) - O
Õ = ($D5) - O
Ö = ($D6) - O
× = ($D7) - X
Ø = ($D8) - 0
Ù = ($D9) - U
Ú = ($DA) - U
Û = ($DB) - U
Ü = ($DC) - U
Ý = ($DD) - Y
Þ = ($DE) - p
ß = ($DF) - b

ç = ($E7) - C
è = ($E8) - e
é = ($E9) - e
ê = ($EA) - e
ë = ($EB) - e
ì = ($EC) - i
í = ($ED) - i
î = ($EE) - i
ï = ($EF) - i
ð = ($F0) - o
ñ = ($F1) - n
ò = ($F2) - o
ó = ($F3) - o
ô = ($F4) - o
õ = ($F5) - o
ö = ($F6) - o
÷ = ($F7) - /
ø = ($F8) - o
ù = ($F9) - u
ú = ($FA) - u
û = ($FB) - u
ü = ($FC) - u
ý = ($FD) - y
þ = ($FE) - p

1.7.24 File Details

You can examine the details of the file you are currently editing using the “Details” dialog. To view file details, select the “File|File Details” option from the menu bar. The dialog divided up into two pates:

Page I – Opening Parameters. This page displays details about the where the file being edited comes from and how the editor was invoked to edit the file. Here is a detailed description of the information:

Raw Command Line. This option shows the Command Line that was passed to XED when it was first executed. When XED is run by another program, the name of the file to edit and other parameters are passed on the Command Line. This option allows you see what parameters were passed and allows you to see any problems.

Command Line Parameters. This section splits the command line parameters onto separate lines so that it is easy to see how the program and Windows divides up the each item.

Full Filename. This option displays the full path and filename. Normally, XED displays the basic filename on the Caption Bar. However, if you have more than one copy of the file on your computer, this is not enough information to be sure that you are editing the correct copy. The Full Filename allows you to verify that you are editing the correct file.

Page II – Property. This page displays intrinsic properties of the file currently loaded into the Editor. The following information will be displayed:

1. Creation Date. This value shows the date and time when file was originally created.

2. Access Date. This value shows the date and time when file was last accessed. For example, this would be the date and time the file was read or opened without doing any writing or modification.

3. Write Date. This value shows the date and time when the file was last written to or modified.

4. File Size. This is the size, in bytes, of the file.

5. Unicode. This checkbox indicates whether the file is written using Unicode or ASCII text. You can also use the checkbox to change the encoding used with file. For example, if the file was originally ASCII, enabling this checkbox and saving the file will caused it to be saved as Unicode.

Page II. This page is used analyze the various aspects of the text in the file. Specifically, it displays counts of characters, words, lines, sentences, paragraphs, and pages in the file. Also, if you enter your own word or phrase in the Utility String box, the program will count occurrences of that string.

1.7.25 Merging Files

One of the few tasks that is easier to do under DOS than in Windows is merging one or more files. In DOS you just use the copy command to merge a series of files:

```
Copy a.txt+b.txt+c.txt d.txt
```

For this reason, XED has a special feature that allows you merge any number of files in any order from any part of the hard drive. To use this feature, select the “File|Copy/Merge Files” option from the menu bar. This will display the “Copy/Merge” dialog box.

Merging Files. The feature works by allowing you to create a list of files that you want to merge. You then specify an output file and the program will copy all the files in the list to the output file. The files are copied in the order they appear in the list and files can come from a mixture of directories and drives.

Select Input File(s) Button. Pressing this button will allow you to select files for the list. Note that you can select multiple files either by holding down the Control or Shift key or by dragging the mouse over several files.

Select Output File Button. Pressing this button will allow you to select a file to which all the input files are merged. As before, the Output file can be in Directory or Drive.

Copy Files Button. Pressing this button will merge all the files in the “Source List” and place the result in the “Destination File.”

Source List. The Source List displays all the files that will be merged together in the Destination File. Files will be merged in the order in which they appear in the list.

List Tools. There are several tools that are used to manipulate the list. They appear as buttons to the left of the list. Here is a description of each tool button:

1. Clear List Button. Pressing this button clears the list by removing all entries.

2. Move Up/Down. Pressing one of these buttons moves the selected item in the list up or down one location. Since the files are merged in the order they appear in the list, these buttons are used to change the order of items in the list.

3. Remove Button. This button removes the selected item from the list.

4. Sort Files Button. This option sorts the files in the list alphabetically by filename.

Options. The following options are available:

Show Full Path Checkbox. To keep the list from being too clutter, the program normally displays only the filenames in the list. Checking this option causes the file and path to displayed.

ASCII CheckBox. Some ASCII files will have Ctrl-Z's at the end of the file as "End-Of-File" markers. At the same time, some editors will stop at the first Ctrl-Z when reading the file. If you merge several files that are terminated with Ctrl-Z, your editor may only display the first file. This options removes Ctrl-Z's as the files are merged.

1.8 Advanced File Commands

XED allows you to work with text files in a variety of ways. Here are complete descriptions of the advanced file operations:

Manipulating Files
Appending Files
File Associations
Integrating With Windows Programs
Backup Files
Large Files
File Access Conflicts

1.8.1 Manipulating Files

Most of the time, you will edit a document using only the simplest file handling commands. This means simply using the "Open" and "Save" commands from the "File" menu. However, sometimes it is useful to do more complex file manipulation. XED has several command-mode commands that can be used to manipulate files.

Here is a list of File Manipulation commands and topics:

Opening and Appending Files
File Associations
Backup Files
Large Files

1.8.2 Appending Text

There are several commands that allow you to manipulate files under the control of Command Line Marcos. Here is a complete description of the commands:

A – Append Command. The Append Command allows you to open a new file and add that text to the contents of the editor. The filename and path follows immediately after the command and should be enclosed in double quotes. Here is a typical Append Command:

```
<Esc>a”c:\animals\cat.txt”<Esc><Esc>
```

The Append Command does not delete any text from the Editor and the new text is always inserted at the end of the buffer.

O – Open and Read. The Open Command discards all text in the Editor and reads the file specified on the command line. The filename and path follows immediately after the command and should be enclosed in double quotes. Here is a typical Append Command:

```
<Esc>o”c:\felines\manx.txt”<Esc><Esc>
```

If the buffer contains modified text, the Editor will prompt you to save the information. The editor will then delete the old text and load the new file.

1.8.3 File Associations

There are several ways that XED can be used in Windows. You can execute XED from Windows start menu and then open a file for the “File” menu. However, it is much easier to use XED by double click on a text file and having XED open it. (Note: you can also configure programs so they use XED as their default text editor instead of Notepad.)

Associating Files. In order for files to be edited by XED when you double click, you must associate XED with various file extensions. You do this by going to My Computer and selecting “View|Folder Options” from the menu bar. Next, select “File Types” tab. This will display a list of all the file types and allow you to add or change how they work.

When you associate a file, you are basically telling Windows what it should do when you double click on the file. For example, if you associate XED with files that have the extension “.TXT”, XED will open any files with this extension when you double click on it.

In order to have XED edit a file when you double click on it, the “Open” command must call XED. To do this, select the file type from the list and then click on the Edit Button. Next, select the “Open” Action and click the Edit Button for it. Finally, “browse” to find the copy of XED you have installed on your computer. The name will be “XEDIT.EXE”.

Long Filenames. Starting with Windows 95, the operating system supports the use of filenames up to 256 characters long. These file names can have “space” characters embedded in them. This can cause a problem for XED since it normally assumes that space are used to separate arguments. To solve this problem, you should put double quotes around the filename part of the argument. In Windows, the filename is usually specified by inserting the following command into the command line:

```
%1
```

By putting double quotes around the command like this: “%1”, will put double quotes around the argument passed to XED when Windows runs the editor.

Other Associations. You can also associate XED with other actions. This is useful for creating alternative actions to the double click. For example, most files with the extension “DOC” are Microsoft Word documents. Occasionally, however, they will be plain text files and if you double click on it, you will open it in Word.

You can get around this by adding a new “Action” to the list for Word Documents. You then associate the Edit actions with XED. This will cause an “Edit” menu item to appear whenever you right mouse click on a file with the “DOC” extension. This will open the document using XED.

Finally, you can also open a file by dragging and dropping it over the XED icon. To do this, you must create a shortcut to XED and put it on the Windows desktop. You can get more information on File Associations and Shortcuts by checking the Windows help files.

1.8.4 Integrating With Windows Programs

Many Windows program allow you to select which editor they use when you do text editing. Normally, these programs use the Windows Notepad, but you can select XED as the editor. XED has several features that allow you to do this. (Note: XED also allow you to associate files with XED so they can be viewed by double clicking.)

Command Line Arguments. Whenever any Windows programs are executed, one or more arguments can be passed to the program. These are called “Command Line Arguments” because they originated with old DOS Command Line. (They are different from XED’s command line, which is used to execute editing macros.)

These Command-Line arguments are used by other programs to control what XED does when it starts. Currently, there are two Command Line arguments.

File Name And Path. If the filename and path of text file is passed to XED on the Command Line, XED will open that file for editing. For example, passing “C:.TXT” would cause the specified text file to be opened.

Line Number. XED supports the option of starting at a particular line number. You do this by adding a switch to the command line arguments that run XED. The switch is “/#” followed by a space and the line number. For example, a typical line number argument would be:

C:MYSTORY.TXT /# 123

This would move the cursor to line 123 of MYSTORY.TXT after the file was opened. The “switch” and line number can appear anywhere on the command line as long as it is separated from the rest of command line by at least one space.

Connecting XED To Other Programs. Many programs allow you to select XED as your editor when working with text. Usually, there is option in the menu of the program that allows you to select a different editor. You just enter the path and filename (XEDIT.EXE) for XED.

Some programs also allow you to specific the filename that XED will work on. Usually this is done through some sort of macro symbol. For example, in Delphi the macro command for the current file being edited is “$EDNAME”. In this case, putting $EDNAME in the command sequence would cause Delphi to send the name file that is being edited on the command line.

Many of these programs can also send a line number to the editor when it opens a file. This enables the editor to go directly to a target line in the document. Again, this is usually done using macro. In Delphi, the macro is “$ROW”. Place this macro in the command sequence instructs Delphi to send the line number of the current file to XED. In order to make this work, you must put the “/L “ in front of the line number so XED will recognize it. For example, the following line show a complete command line macro sequence for Delphi:

/# $ROW $EDNAME

Long Filenames. Starting with Windows 95, the operating system supports the use of filenames up to 256 characters long. These file names can have “space” characters embedded in them. This can cause a problem for XED since it normally assumes that space are used to separate arguments. To solve this problem, you should put double quotes around the filename part of the argument.

Debugging Command Line Parameters. Since it is often difficult to know what parameters a third party program may be passing to XED, there is an option to view those parameters . This option allows you to see and adjust the third party program to have XED respond correctly.

1.8.5 Backup Files

When you save a file in XED, it creates a new copy of this file that contains any changes. XED normally creates backup files while it works. This protects you from losing data. XED never overwrites the input file. Instead, the original input file is renamed to a .BAK file, and the new text is written to the original filename. This way, there is always an unmodified copy of your original file.

1.8.6 Large Files

XED holds text as array of lines. With virtual memory, the maximum number of lines is be 1-gigabyte of lines (4-gi / 4-byte per pointer). Since, each line itself could hold 4-gigabytes, the theoretical maximum size could be 2.1e12 characters. Of course the actual amount depends on the number of lines. The maximum for a single line of text is 4-Gigabytes. For two lines of text it would 8-gigabytes and so on.

In reality, the amount of text that can be loaded will depend on the capabilities of the virtual memory system and how much real memory and disk space there is. On a system with 500-megabytes of RAM, I was able to load a 128-megabyte text files with no problem. I suspect that things would slow down as I approach or exceed the amount of physical memory.

1.8.7 File Access Conflicts

XED can open files that are already open by other programs. If two copies of XED have the same file open, either one can save text to the open file.

1.9 New Features

The Windows version of XED adds several features. Many of them take advantage of the increased power and resources of the Windows environment. Here is list of the new features that only exist in the Windows environment:

1. Mouse and Scoop Highlighting. XED now supports mouse highlighting and highlighting of the text that is beginning pulled into the scoop. For more information about highlighting, click here.

2. Large Undo/Redo Buffer. XED now has the capability of saving and undoing the last 32,000 changes. Also, if you do series of “Undos,” you can also “Redo” the undone commands. For more information Undo/Redo, click here.

3. Buffer Size. The buffer size limited only by virtual memory under Windows. On a 500 megabyte

system, running Windows XP, I have loaded files as a large as 350-megabytes.
free memory.

4. The Command Line. The Command Line is now completely editable like the edit screen itself. This means that the mouse, arrow keys, clipboard all work with the command line. For more information on the command line, click here.

5. Reopening Recently Edited Files. XED remembers the last five files that it opened. It displays these files in menu bar under “Files|Reopen.” This allows you to reopening any of these files with just a few mouse clicks.

6. Save In Place. XED has a new feature that allows you to save the current state of the file without losing your place in the file. The feature works with multiple buffer files. For more information saving files, click here.

7. Text Ruler. XED now has text ruler that allows you to visually set the position of the Margins and Paragraph Indent positions. For more information about the Ruler, click here.

8. Clipboard Support. XED now supports the cutting, copy and pasting to and from the Windows Clipboard. For more information about the Clipboard, click here.

9. Find and Replace. XED now supports the option of using the Windows Find and Replace dialogs. This makes it easy to find and selectively replace text. For more information about Find and Replace click here.

10. Colors and Fonts. XED now allows you to set the size, face and style of screen fonts. You can also set all the colors for fonts, background, screen highlights, control characters and color of the screen display fonts. For more information about the Fonts, click here. For more information about Screen Colors, click here.

12. Saving Defaults. XED now allows you to save 29 different settings as operating defaults. This allows you to completely customize the operation to the editor. For more information about Defaults, click here.

13. Pair Matching. XED now allows you to find the matching “end” symbol for any “begin-end” pair. For more information about the Pair Matching, click here.

14. Options. XED allows you to set a range of options that control the way the editor works. These include what tools appear on the screen, how the screen scrolls and other options. For more information about the Options and Setting, click here.

15. Block Case Changing. XED now allows you to change the case of a highlighted block of text. The options include Upper Case, Lower Case, Title Case, Sentence Case, and Toggle Case. For more information about Case Shifting, click here.

16. Block Indent. XED now allows you to indent or unindent a highlighted block of text. For more information about Block Indents, click here.

17. Margin Presets. XED now stores four margin settings that can be easily recalled by simply Right Mouse Button clicking on the Text Ruler. For more information about Margin Presets, click here.

1.10 Compromises, Caveats, and Differences

In writing this version of XED I have tried to retain the best features of XED while incorporating useful features from Windows-style editors. Generally, this has worked well, but there are some places where a few minor compromises, changes and even improvements were required.

1. Alt Characters. Windows makes extensive use of the Alt-Key in combination with other keyboard characters. In particular, Windows uses the Alt-Key in combination with various alphabetic characters to highlight items on the menu bar. For this reason, it would have been difficult to use Alt-Key commands from XED. As a result, Control-Shift has been used to replace the Alt command. Note that this only applies to Alt combined with Alphabetics. Alt still work fine with function keys and non-printing keys.

Index