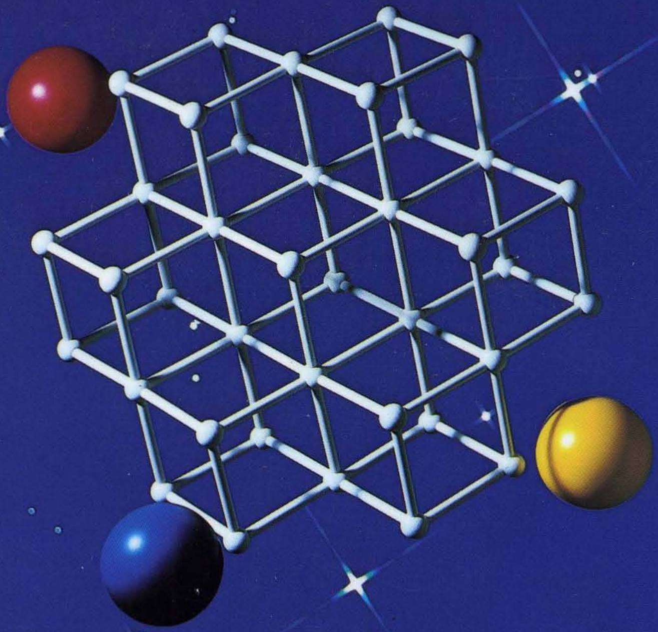
LOGITECH™ MODULA-2

VERSION 3.0



POINT EDITOR

POINT™ EDITOR

Version 1.5

User's Manual

LOGITECH, Inc. makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability and fitness for a particular purpose. The information in this document is subject to change without notice. **LOGITECH, Inc.** assumes no responsibility for any errors that may appear in this document.

From time to time changes may occur in the filenames and in the files actually included on the distribution disks. **LOGITECH, Inc.** makes no warranties that such files or facilities as mentioned in this documentation exist on the distribution disks or as part of the materials distributed.

Third Edition, September 1987

This User's Manual applies to *POINT Editor, Version 1.5*.

Document #:	LU-UD-008-1
Initial issue:	February 1984
Current revision:	August 1987
Printed:	August 1987

Trademarks

LOGITECH and *POINT* are trademarks, and *LOGIMOUSE* is a registered trademark of **LOGITECH, Inc.**

IBM is a registered trademark of **International Business Machine Corporation.**

Microsoft, *MS*, and *MS-DOS* are registered trademarks of **Microsoft Corporation.**

Intel is a registered trademark of **Intel Corporation.**

Hercules is a trademark of **Hercules Computer Technology.**

Hewlett-Packard, *HP*, and *LaserJet* are registered trademarks of **Hewlett-Packard Corporation.**

Byte is a registered trademark of **McGraw-Hill, Inc.**

AT&T and *UNIX* are registered trademarks of **American Telephone and Telegraph Corporation.**

COMPAQ is a registered trademark of **Compaq Computer Corporation.**

LOGITECH SOFTWARE LICENSE AGREEMENT

THIS DOCUMENT IS A LEGAL AGREEMENT BETWEEN YOU, THE LICENSEE, AND LOGITECH, INC ("LOGITECH"). BY USING THIS PROGRAM, YOU ARE AGREEING TO BECOME BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, PROMPTLY RETURN THE DISK PACKAGE AND THE OTHER ITEMS THAT ARE PART OF THIS PRODUCT IN THEIR ORIGINAL PACKAGE, WITH YOUR PAYMENT RECEIPT (THE "RECEIPT"), TO LOGITECH FOR A FULL REFUND.

In consideration of payment of the License Fee, which is a part of the price evidenced by the Receipt, LOGITECH grants to the Licensee a nonexclusive right, without right to sublicense, to use this copy of this LOGITECH Software on a single Computer at a time. LOGITECH reserves all rights not expressly granted, and retains title and ownership of the Software, including all subsequent copies in any media. This Software and the accompanying written materials are copyrighted. You may copy the Software solely for backup purposes; all other copying of the Software or the written materials is expressly forbidden.

As the only warranty under this Agreement, and in the absence of accident, abuse or misapplication, LOGITECH warrants, to the original Licensee only, that the disk(s) on which the Software is recorded is free from defects in materials and workmanship under normal use and service for a period of ninety (90) days from the date of payment as evidenced by a copy of the Receipt. LOGITECH’S only obligation under this Agreement is, at LOGITECH’S option, to either (a) return payment as evidenced by a copy of the Receipt or (b) replace the disk that does not meet LOGITECH’S limited warranty and which is returned to LOGITECH with a copy of the Receipt. THIS WARRANTY GIVES YOU LIMITED, SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHER RIGHTS, WHICH VARY FROM STATE TO STATE.

THE SOFTWARE AND ACCOMPANYING WRITTEN MATERIALS (INCLUDING THE USER’S MANUAL) ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, EVEN IF LOGITECH HAS BEEN ADVISED OF THAT PURPOSE. LOGITECH SPECIFICALLY DOES NOT WARRANT THAT THE OPERATION OF THE SOFTWARE WILL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL OR INCIDENTAL DAMAGES ARISING OUT OF THE USE OF OR INABILITY TO USE SUCH PRODUCT EVEN IF LOGITECH HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, SO THE ABOVE LIMITATION MAY NOT APPLY.

Table of Contents

Introduction 1

Installation 7

1. Begin the Tutorial 11

Chapter 5 How POINT is Displayed 81

Chapter 6 POINT Commands 97

Appendices

POINT EDITOR

Introduction

This introduction gives you an overview of the *POINT* Editor, and tells you how to read the manual, as well as providing some information about **LOGITECH** products and product support.

How the POINT User's Manual is Organized

This manual assumes that you are familiar with the basics of *DOS* and with basic programming concepts and terminology.

This manual includes:

- Introductory information with system requirements and installation instructions.
- A step-by-step tutorial through the *POINT Editor* in Chapters 1 and 2.
- A complete reference for the *POINT Editor* in Chapters 3 through 6.
- Supplements and **Quick Reference Guides** in the Appendices
- An index.

If you are a beginner, work through the tutorial to get started using the *POINT Editor.* Then consult the reference chapters for more detailed information.

How to Read This Manual

The following conventions are used in this manual:

Keys to be pressed, look like this:

Y   Esc   ↵

Control sequences or characters entered with a **Control** or **Shift** key, look like this:

Ctrl-C   Ctrl-Break   ⇑Shift-F2   Alt-X

Keys from the **Numeric Keypad** are shown like this:

↑   ↓   ←   →

PgUp   PgDn   +   −

Keyboard input for the ***DOS*** Command line is in upper case and looks like this:

M2L ↵

Mouse buttons used are based on the ***LOGITECH*** standard, and use three buttons, e.g,

■ □ □ means press the left mouse button,
□ □ ■ means press the right mouse button, and
□ ■ □ means press the middle mouse button.

■ ■ refers to the middle button as implemented on a two button mouse.

File names look like this:

M2L.EXE

DOS commands and statements look like this:

PATH, COPY

Product names look like this:

MS DOS, LOGITECH Modula 2

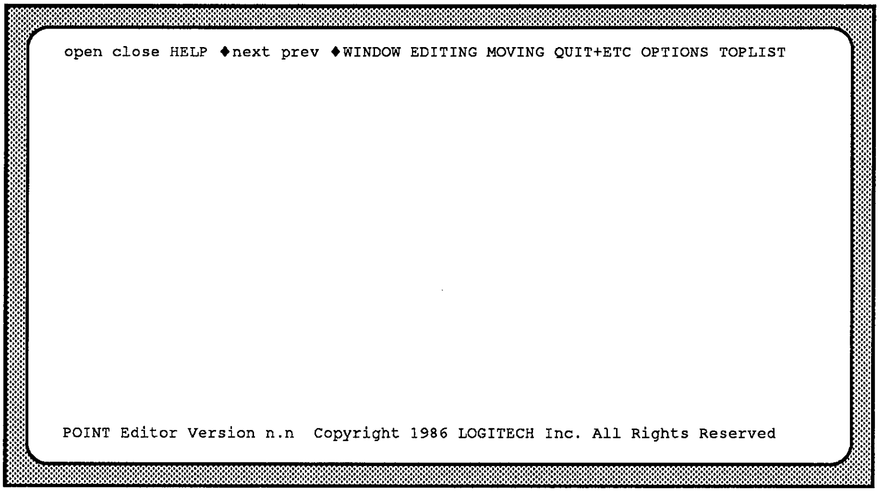
Screen output and some listings look like this:

```
Program Not Found
```

Program source code looks like this:

```
IF condition THEN
   statement6;
ELSIF condition THEN
   statement7;
ELSE
   statement8;
END;
```

Sample Screens look like this:


open close HELP ♦next prev ♦WINDOW EDITING MOVING QUIT+ETC OPTIONS TOPLIST

```
POINT Editor Version n.n  Copyright 1986 LOGITECH Inc. All Rights Reserved
```

LOGITECH Policies and Services

LMIS

We know that effective communication with our customers is the key to quality service. Therefore we have set up the **LMIS** (**L**OGITECH **M**ouse **I**nformation **S**ervice), an electronic bulletin board where you can contact us at *your* convenience. To reach the **LMIS**, dial:

(415) 795-0408

using a 300, 1200 or 2400 baud modem.

The menu of available options is self explanatory.

BIX

LOGITECH also sponsors an electronic conference on ***BIX***, the **B**YTE **I**NFORMATION E**X**CHANGE system from ***Byte*** magazine. If you have access to ***BIX***, join us in

conference **LOGITECH**,

and communicate with us there.

Getting Help through the Hotline

You should rely on your manual or your dealer to answer questions about using your package. If you do encounter a technical problem with your package, our Technical Support Specialists will be glad to help you.

We ask you to follow these steps before you call or write.

- Read the section of the manual that describes the procedure you are trying to perform.
- If the problem relates to your software, check to make sure that the software is properly configured.

If, after following these steps, you are still not able to solve the problem, give us a call at *(415) 795-0427*, or write to us. If you write, please include your daytime phone number and the best time to reach you. Also, please add "***Attn: Technical Support***" somewhere on the envelope.

We want to help you make the most effective use of your package.

Other LOGITECH Products

At **LOGITECH** we pride ourselves on technical excellence and advanced engineering. In addition to the *POINT Editor,* we offer these fine products which we believe to be the most advanced in their product categories.

LOGITECH Modula-2 Development System

The ***LOGITECH Modula-2 Development System*** gives you the most powerful software development environment available for your PC:

- The ***LOGITECH Modula-2 User's Manual*** with tutorials, a Compiler that generates standard .OBJ files, a Symbolic Post-Mortem debugger, and a complete listing of the ***LOGITECH Modula-2*** libraries.
- The ***LOGITECH Modula-2 Toolkit*** includes the enhanced ***LOGITECH Linker***, a Symbolic Run-Time Debugger, and the perfect combination of utilities to optimize both your development environment and your ***Modula-2*** code.
- The ***LOGITECH Turbo-Pascal To Modula-2 Translator.***
- A *VAX/VMS version* of ***LOGITECH Modula-2.***

Site licenses are available for all ***LOGITECH Modula-2*** products.

The **LOGITECH C7 Mouse**

The **LOGITECH** *C7* **Mouse** connects to a serial port in your computer. It needs no pad and no external power supply.

The **LOGITECH Bus Mouse**

The **LOGITECH Bus Mouse** is equivalent to the ***LOGITECH C7 Mouse***, except that it is connected to a Bus Board which you insert in your computer. It needs no pad and no external power supply.

For additional information, or to order these products, call the **LOGITECH** sales office toll-free from anywhere in the **continental U.S.** at **(800)231-7717,** or in **California**, call **(800) 552-8885.**

Notes:

Installation

This chapter tells you how to install the *POINT* Editor on your system and create an optimal environment for your editing tasks.

What You Need

To install the *POINT Editor* you need:

An *IBM PC/XT, AT*, *PS/2* or compatible computer, with

- **256 K RAM** memory (*POINT Editor* uses **140 K**)
- Two (2) **360 K** disk drives, or a **hard disk** and at least one (1) **360 K** disk drive.
- *PC DOS* or *MS DOS 2.0*, or above.

- A **LOGITECH mouse** is definitely recommended.
- If you are not using a mouse, refer to Appendix E for instructions on keyboard emulation of mouse actions.

IMPORTANT

READ.ME on one of your **LOGITECH** disks has information about this version of the *POINT Editor* that was not available when the manual was prepared.

POINT Files on Disk

The following *POINT* files are on your **LOGITECH** distribution diskette.

* PT.EXE The **executable** file for the editor.

* PT.INI The **initialization** file.

* PT.MSG The **message** file, contains a one line description for each command.

* PT.HLP The **help** file, gives an extensive summary and detailed descriptions of commands and menus.

* indicates files needed to run POINT.

Additional files issued with the *POINT* Editor are used as follows:

COLOR.INI The **color** initialization file. If you have a color monitor, copy COLOR.INI to PT.INI.

MONO.INI The **monochrome** initialization file, this file as well as the original PT.INI is set up for monochrome monitors (also for black and white monitors running with the color card).

If you have been using a color monitor and must now use a monochrome monitor, copy this file back into PT.INI.

PTEXPERT.INI The **expert** user initialization file. Copy this file into PT.INI when you are more familiar with the ***POINT Editor***.

Alternate .INI files can be copied to the name PT.INI for customizing your ***POINT*** system. See **Chapter 4**.

READ.ME This file contains information that was not available when *POINT* documentation was prepared.

Installation Procedure

NOTE

Before you install your software to either floppy drive or hard disk system, we strongly recommend that you take a minute to:

1) Put Write-Protect tabs on all your *LOGITECH* diskettes, and

2) Use the **DISKCOPY** and **DISKCOMP** commands from your *DOS* files to back up your diskettes. Then put your original diskettes in an archival area and use the copies for all installation.

3) Prepare formatted diskettes with readable labeling, before you copy the the files in the Installation procedure which follows.

Hard Disk System

If you have a hard disk, run the INSTALL program from **drive A**. It will transfer the appropriate files to the directory you specify, or to a default directory.

Floppy Disk System

For a floppy diskette system, format a target diskette with your *DOS* system for **drive B**.

Then put the **copy** of your **LOGITECH** diskette into **drive A**, and type,

```
A:INSTALL ↵
```

Configure Your System

Set File Handles

You can edit several files at once. ***POINT*** lets you include up to 20 file names on the **command line** when you invoke ***POINT***. A **file handle** is needed to work with each file. If you plan to edit several files at the same time, be sure you have enough file handles.

Put this line in your CONFIG.SYS file:

```
FILES=30
```

This gives you **30** file handles instead of the default of **8**. The resident part of ***DOS*** is only increased by 40 bytes for each file above **8**. This is more than enough for most purposes.

Load Your Mouse Driver

Before you run ***POINT***, if you have a mouse, load your mouse driver. Use MOUSE.COM or MOUSE.SYS. Refer to your mouse manual for instructions.

If you don't have a mouse, refer to the **non-mouse** equivalent tables in Appendix E.

POINT and Your Monitor

If you have a monochrome or ***Hercules*** compatible display, leave PT.INI as it is on the diskette or in the current directory.

If you have a color display, copy the COLOR.INI file to PT.INI on the diskette or in the current directory.

If you have a color graphics adapter you may see display interference (snow) on the screen. Press PgDn and PgUp to check for display interference on the screen. If you see display interference, reset the **`videoMode`** option. To reset the **`videoMode`** you will have to change one parameter in the PT.INI file. Refer to **Section 4.3**.

Chapter 1
Begin the Tutorial

Welcome to the *POINT* tutorial.

When you finish this chapter you will know how to load the *POINT* program files, how *POINT* screens and menus work, and how to manipulate windows and files with the *POINT Editor*.

If you haven't done the setup procedures, go back now and configure your system so you can understand what follows.

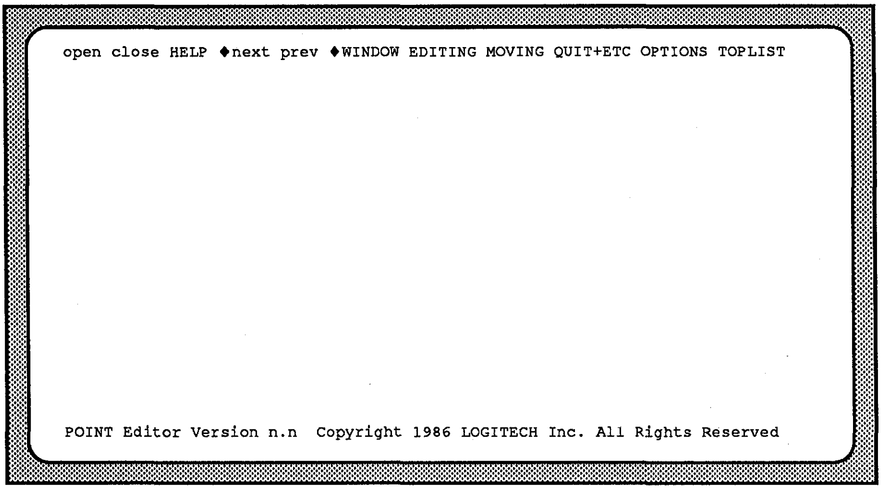
Remember to include the POINT sub-directory in your PATH statement and make sure you are in the POINT sub-directory before you begin this tutorial.

1.1 Load the POINT Editor

To load the *POINT Editor*, type:

PT ↵

You will see the initial *POINT* screen. The top line of the screen is the main menu, the body of the screen is the work area (for loading and editing files), and the bottom line (when you first call the *POINT Editor*) displays the version and copyright notice:


open close HELP ♦next prev ♦WINDOW EDITING MOVING QUIT+ETC OPTIONS TOPLIST

```
POINT Editor Version n.n  Copyright 1986 LOGITECH Inc. All Rights Reserved
```

NOTE

To quit the *POINT Editor* and return to *DOS*, press Alt-Q

Since this is a tutorial, don't exit just yet. If you have exited, please re-start the editor. Refer to <u>Section 6.8</u> for more information on `quit` commands.

1.2 The Parts of the Screen

When the *POINT Editor* starts, it creates a screen with:

- The main menu at the top of the screen;
- A blank work area where one or more windows can be opened and edited;
- A comment line (with the copyright notice) at the bottom of the screen.

NOTE

The comment line alternates between simple help reminders and command prompts. It changes when you move the cursor to the top line of the screen. Occasionally it disappears entirely.

Move the blinking cursor along the main menu at the top of the screen. Observe how the copyright notice disappears from the comment line and is replaced by a short description of commands.

1.2.1 Cursor, Selection, and Insertion

There are three primary primary functions on your *POINT* screen:

Selection cursor — A **blinking square** that can be moved by the mouse — or, if you have no mouse, by keypad cursor keys — to either select a new insertion point, or so select a command outside the text window. The new selection or command is then implemented with [■ □ □] (and sometimes also with [□ □ ■]).

Selection — An **area of text that is highlighted** for purposes of manipulation, or for indicating the insertion point.

Insertion point — The **area at the beginning of the selection** at which new characters appear from your keyboard, or as the result of a move or copy operation from another area of text (from the same window, or even from a different file in another window).

One last word here: the insertion point is set either to **add characters** at the insertion point (`OverType=0`) or to replace existing text with new text (`OverType=1`).

1.2.2 The Main Menu

The **main menu** is on the top line of the screen. It contains **pull-down menus** and commands. Menus, such as `WINDOWS`, are in upper case, and commands, such as `open`, are in lower case.

Each **pull-down menu** contains a list of commands and/or menus.

Each **command** presents choices or prompts, or performs a function.

1.2.3 The Work Area

The work area (the large blank area) of the screen is where you load and edit files.

1.2.4 Comment line

The comment line (the bottom line) displays one line comments, explanations, or command prompts for the pull-down menus and commands that you use.

1.3 The HELP System

The comment line offers a simple level of help. For additional help, move the mouse cursor to the `HELP` menu on the top line and click ■ □ □ . A help screen will appear which gives details on the last command executed or highlighted.

You can return to your editing session from this help screen, or get additional help. To exit the help system, either click □ ■ □ or press the Spacebar , Esc , or ↵ . To access the main help menu, press PgDn .

From the main help menu you can get additional help screens. Use the mouse cursor to pinpoint the subject you need. Then click the ■ □ □ mouse button, or press ↵ . A help screen with additional information will appear.

Some help screens let you access additional help screens with specific information about particular subjects. You can access these screen by pressing a letter or key from the `More info` column which corresponds to the subject matter you choose.

PgDn or □ □ ■ gets the next help screen, and PgUp or ■ □ □ gets the previous help screen.

If there are no more related help screens, PgDn takes you back to the main help menu. From this menu you can access any of the help screens, or exit the help system.

1.3.1 Yes/No Questions

When answering yes or no questions, either `Y` (yes) or `N` (no) is shown as a default answer. You can accept the default with the ↵ key. Or, you can press Y or N as an answer, without pressing ↵ .

You can also use the mouse to answer Yes/No questions: simply move the mouse cursor to the bottom of the screen. Then press ■ □ □ for **Yes** or □ □ ■ for **No**. The mouse cursor must be on *the bottom line* to be able to answer yes or no.

1.4 Look at Menus

The following exercises show screen layout and available commands.

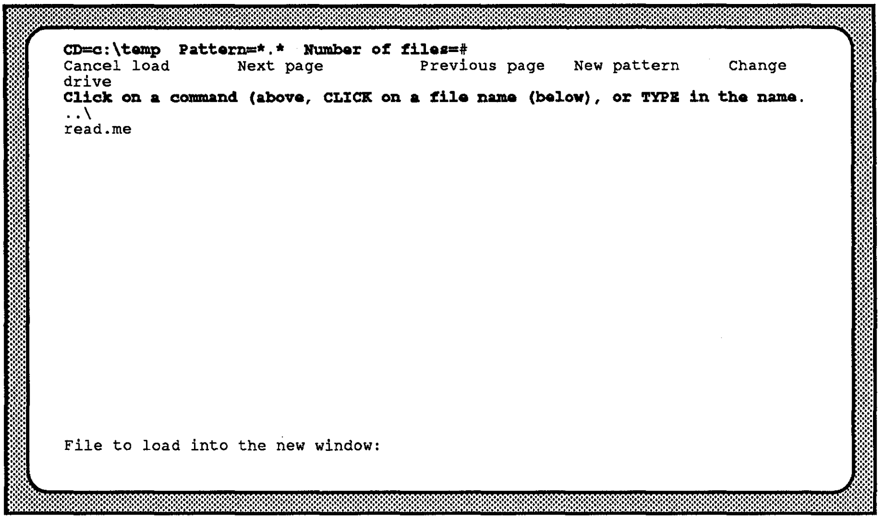
1.4.1 Open a File

Let's explore menus and screen options by opening a window and loading a file.

Step 1: Put the cursor on **open** on the main menu. Now click [■ □ □] twice.

A screen appears, with:

- Directory and file information on the top line
- Additional commands on the second line
- A list of the files in the current disk drive and directory
- A prompt which lets you enter a file name from the keyboard.



```
CD=c:\temp  Pattern=*.*  Number of files=#
Cancel load       Next page          Previous page   New pattern     Change
drive
Click on a command (above, CLICK on a file name (below), or TYPE in the name.
..\
read.me








File to load into the new window:
```

Step 2: Place the mouse cursor on **read.me**. Click [■ □ □] .

The file you just selected now appears inside a window in the work area. Now you can explore the rest of the the *POINT* system. For this tutorial we will use this READ.ME file from the POINT sub-directory.

1.4.2 Explore a Pull-down Menu

The following steps show you how to choose a command from a pull-down menu.

Step 1: Move the mouse cursor to the top line of the screen and put the it a menu. **`WINDOWS`**, **`EDITING`**, **`MOVING`**, **`QUIT+ETC`**, **`OPTIONS`**, and **`TOPLIST`** are menu names.

Step 2: Press and hold the ■ □ □ mouse button on the menu name you have selected. This pulls down a menu under the selected menu name.

Step 3: While pressing ■ □ □, "drag" the cursor into the pull-down menu and down the screen.

Menu items are highlighted as you move the cursor inside the menu. Notice how the comment line changes to reflect the highlighted command inside the menu.

Step 4: Move the cursor outside the menu and release the button. This releases the menu without executing a command.

NOTE

To choose and execute a menu command, release the button when the desired command is highlighted.

1.5 Create a New Window

In this exercise you will create a new window, and load a file into that window. This procedure is similar to the `open` command, but is done from the `WINDOWS` menu.

Leave the READ.ME file on the screen.

Step 1: Select `WINDOWS` and then drag the mouse cursor down the screen to highlight `New Window`. The following message appears on the comment line:

```
Press the left mouse button at one corner.
```

Step 2: Put the mouse cursor in the middle of the screen and hold down the [■□□]. Move the cursor to the lower right screen corner. An elastic window border follows the mouse cursor until you release the button. You have just defined the position and size of the window on the screen.

NOTE

If you double click at the upper left window corner, the lower right window corner goes automatically to the lower right corner of the screen. The new window then covers the current window. Don't worry: the READ.ME file is still there. Go on with the remaining steps. Later you will see how to handle overlapping windows.

Once you choose the two corners, you are shown a screen with a selection of file names from the current directory.

Step 3: Select a file to load into the window by clicking on it with [■□□], or by typing the file name on the keyboard at the comment line prompt.

For this tutorial, type in BRAND.NEW from the keyboard. As you type in the characters they appear at the end of the comment line. When the name is complete, press [↵].

Step 4: You return to the editing screen and are asked to confirm the file creation with a 'y'.

When you type [Y], a window appears with the borders you defined in Step 2.

1.5.1 Look at the Parts of the Windows

Take a minute to look at the windows you now have on the screen.

> NOTE
>
> If and ONLY if your windows are overlapping, go to **Section 1.9, Reshape a Window**. Follow those procedures to resize the overlapping window. Then return here and continue.

Each window has a banner line, an elevator, and a border.

Each banner line (at the top of the window) has:

- The *name of the file* in the window.
- The *line and column ranges* in that window.

Each window elevator (in the left border) shows:

- *Where* in the file you are currently looking.
- *How much* of the file is being displayed.

The window border shows which is the active window:

- The current or active window has a *double line* border. (See Section 5.2.1)
- The active window is affected by ***POINT*** commands.
- An inactive window has a *single line* border.

Put the mouse cursor on the READ.ME window. Click (■ □ □). The READ.ME window is now active.

Toggle between the two windows on the screen by clicking on the banner line of each window. As you do this, see how the borders change.

1.5.2 Close the Window

When you finish editing a file, clear it from the working area.

Step 1: Make the READ.ME window active.

Step 2: Now, close the window in one of three ways:

- Use ■ □ □ to pull down the **`WINDOWS`** menu; release it on **`Close Window.`**
- Click ■ □ □ on **`close`** on the top line.
- Press Alt-C.

Each method closes the active window and clears it from the screen. The BRAND.NEW window is left on the screen, and automatically becomes the active window.

If the window to be closed has been edited, a comment line prompt shows the drive, the path name if any, and the file name, along with the option to save or discard edits.

The default is Y to save. For this tutorial press N.

1.6 Open Multiple Windows

Multiple windows are useful when editing several files. Now you will open multiple windows and see how they exist simultaneously on the screen.

Step 1: Select `WINDOWS` with the mouse cursor and choose `New Window`. A message on the comment line asks you to:

```
Press the left mouse button at one corner
```

Step 2: Press and hold [■ □ □] near the upper left corner of the screen. An elastic window border will follow the mouse cursor until you release the button at the lower middle section of the screen. When the list of file names appears, type:

READ.ME [↵]

Step 3: Repeat Step 1 and Step 2; only this time define the window, starting from the upper right hand corner.

You now have *three* windows on your screen: *two* loaded with the file READ.ME and *one* with BRAND.NEW.

Step 4: If only two windows are visible on the screen, put the mouse cursor on `TOPLIST` and hold down the [■ □ □] mouse button, or press [■ □ ■] or [■ ■] or [□ ■ □] to view the list of files loaded in windows.

1.7 The TOPLIST Menu

1.7.1 The Active Window

When you have several windows on the screen, you need to find the **active** window, i.e., the window affected by menu and keystroke commands. Only one window can be active. It has a double-line along the borders and in the banner. Non-active windows have single line borders and use blank fill characters in the banner.

To set the active window:

[■ □ □] anywhere along the banner of any window.

1.7.2 Overlapping Windows

POINT lets windows overlap and obscure other windows. Think of the windows as sheets of paper on the screen. The top page can be seen completely and the lower pages may be partly or wholly hidden by windows above them. You can choose windows by clicking [■ □ □] on any window corner.

This moves a window to the top of the stack, but does not change its position on the screen. If the window is already the top window, this sends it to the bottom of the stack.

[Alt]-[B] also sends the active window to the bottom of the `TOPLIST` stack.

`TOPLIST` makes it easy to switch between windows. `TOPLIST` also makes it easy to keep track of which files you are actually using and where they are in the work area.

From anywhere below the top line of the screen press [□ ■ □] for the `TOPLIST` menu.

To select an active window with `TOPLIST`:

Step 1: Press [□ ■ □]. This invokes the `TOPLIST` menu.

Step 2: Highlight the file name you want with the mouse cursor, and release the mouse button. The name you highlighted becomes the active window and moves to the top of the stack of windows. This is called **Topping**.

1.8 Move a Window

Sometimes you must move a window to uncover another window, or make room for an additional window:

Step 1: Select a window.
Press and hold [□ □ ■] on the banner line of the window. A shadow version of the window border follows the mouse.

Step 2: Move the window against the bottom edge of the screen and watch the border get smaller. The border expands to its original size as you move away from the bottom of the screen.

Step 3: Release [□ □ ■] while the shadow window is smaller, to reduce the size of the window. If the final size of the window is too small, the move will be canceled.

> NOTE
>
> A window can also be moved on the screen simply for better viewing, without reducing its size.

1.9 Reshape a Window

Sometimes you need to make a window larger or smaller. Here are three ways.

1.9.1 Resize a Window

Step 1: Activate the window you want to resize, as discussed in Section 1.9.1.

Step 2: Select `Resize Window` in the `WINDOWS` menu. A prompt in the comment line now requests two opposite corners of the window.

Step 3: With the cursor anywhere on the screen, press and hold [■ □ □].
An elastic window now follows the cursor to the desired size and shape. Then release the button: the window and its contents will take the shape, size and position you have just specified.

> NOTE
>
> You can specify either pair of opposite corners for the window (not just the upper left and lower right corners).

1.9.2 Stretch a Window

Stretch is faster than `Resize` for repositioning windows.

Step 1: Put the mouse cursor on any window corner and press the [□ □ ■].

Step 2: Still holding down the [□ □ ■], move the mouse cursor. An elastic border follows the mouse cursor until you release the button.

Step 3: Repeat Step 1 and Step 2 using a different corner.

1.9.3 Zoom a Window

`Zoom Window` toggles between a full-screen active window and a multi-window screen.

Step 1: Select the smallest window to `Zoom` and make it the active window.

Step 2: Press [Alt]-[Z], or select `Zoom Window` from the `WINDOWS` menu.
Repeat to unzoom the window.

1.10 View a File

A window often shows only part of a text file. Scrolling "moves the window" vertically or horizontally over the file so you can see different areas of the text.

1.10.1 Scroll With the Mouse

To scroll vertically:

Step 1: Place the mouse cursor on the *left border* of the window.

Step 2: Click [□ □ ■] to scroll down, and [■ □ □] to scroll up.

The farther down the left border the mouse cursor, the further the window scrolls. More precisely, [□ □ ■] moves text beside the mouse cursor to the top line in the window. [■ □ □] moves the top line to where the mouse cursor is on the left margin.

Watch the the banner line change and the elevator move along the left window border.

To scroll horizontally:

Step 1: Place the mouse cursor on the *bottom border* of the window.

Step 2: Click [□ □ ■] to scroll right and [■ □ □] to scroll left.
Notice the changes in the columns indicator on the banner line.

1.10.2 Thumb with the mouse

Thumbing is like holding a book or magazine so you can flip through it with your thumb. Similarly, you can use the mouse to move the window to any part of the file by *thumbing*.

Imagine that the left window border (but not the corners) is the whole file. Now move the mouse cursor to the *center of the left border*. Press [□ ■ □] : the elevator moves to the *center* of the border and the window jumps to text in the the *middle of the file*. For text at the end of the file, press [□ ■ □] near the bottom of the window's left border.

When text in the window is out of range on either left or right border, press [□ ■ □] anywhere along the bottom border to thumb horizontally back and forth.

NOTE

Don't try to scroll or thumb window corners. Corners have different commands.

1.10.3 Scroll with Cursor Keys

[PgUp] or [PgDn] also scrolls the window up or down.

[Ctrl]-[PgUp] moves a window to the beginning of the file.

[Ctrl]-[PgDn] moves a window to the end of the file.

NOTE

With [PgUp] or [PgDn] a window is scrolled *n-2* lines.

This means a window with *14 lines* moves *up* or *down 12 lines*, with *2 lines* of reference to the previous screen.

1.11 Save Files

Editing space depends on the *number of changes* you make to the files you edit, rather than the *size of the files* edited. When editing space gets crowded, the `SAVE` message appears in front of the file name in each window.

When `SAVE` appears, save edited files to regain editing space. If you don't save the files as soon as you see this message, you risk losing your edits.

Use `Save File` from the `WINDOWS` menu or Alt-S .

1.11.1 The Save As ... Option

Sometimes you want to keep the changes in a file separate from the original file. In this case, save the changed file *under a different name.* This makes it possible for you to have both the current file under a different name and the original file under its old name.

Thus, if you make changes to READ.ME and want them saved separately, do this:

Step 1: Make one of the READ.ME windows active by putting the cursor on the banner line and clicking ■□□.

Step 2: Choose `Save as...` from the `WINDOWS` menu or press Alt-W . The comment line will display:

Step 3: Type READ.YOU ↵ .
The file in the active window will be saved as READ.YOU.

Step 4: Close the current window with Alt-C .

Step 5: Open a new window, using `open`, or `WINDOWS/New Window`.
READ.YOU will appear in the list of files in the directory.

Step 6: Load READ.YOU by clicking ■□□ on the name from the list, or by typing it on the comment line prompt. READ.YOU now appears in the banner line of the new window.

You now know how to create and manipulate windows with *POINT*. The next chapter will show you how to edit and manipulate files.

Notes:

Chapter 2
Edit With POINT

This chapter lets you edit and manipulate files with keyboard commands and with the `EDITING` menu.

Using the files from the previous chapter, open a window and load a file. Use more than one window if you desire.

2.1 Insert New Text

When you type, characters are inserted immediately to the left of the selection highlight, which then is "pushed ahead" of the cursor.

To insert a new line, press ↵ .

To start a blank line, select the **end-of-line character** of the preceding line. Find it by clicking the ■□□ **anywhere to the right** of the last visible character on the line. The **rectangular highlighted blank** contains the *Carriage Return* character. Now press ↵ .

2.2 Select Text

Text can be deleted, moved, copied in and between files, or it can be temporarily stored in a scrap buffer. The first step for any of these actions is to select the text.

2.2.1 Select Characters

Point at text with the mouse cursor and click [■ □ □].

If you press and hold [■ □ □] while you move the mouse cursor (in either direction) you can select any number of characters. If you extend the selection, once the [■ □ □] mouse button is released, the selection will start again from where the mouse cursor is currently positioned.

To extend the current selection, hold down the [□ □ ■] and move the mouse cursor. The selection is extended leaving the originally selected text highlighted.

NOTE

There is a non-displayable character at the end of each line that is "displayed" as a blank. You can see it only when it is part of a highlighted selection. (Select over several lines so you can see it.)

This is the end-of-line character. It represents the Carriage Return character or the [↵] key that ends each line. If you delete the end-of-line character, that line fastens itself immediately to the first character of the following line.

2.2.2 Select Characters, Words, and Lines

A selection extends from character to word to line, and back again to character.

Select a single character (in a word) with the blinking mouse cursor and click the [■ □ □] button. Click again to extend the selection from *character* to *word.* Click again to extend the selection from the *word* to the *entire line.* A final click returns you to *the original character you selected.*

To extend the selection by character, word, or line, hold down the mouse button on the last click and move the cursor up or down in the file.

Step 1: Click [■ □ □] on any character in the text. The *character* is highlighted.

Step 2: Click [■ □ □] on the same character. The whole *word* is now highlighted.

Step 3: Click and hold down [■ □ □] on the same character. Now *the entire line* is highlighted.

Step 4: Move the cursor up or down to extend the selection. Notice how the selection is highlighted line by line.

NOTE

If you hold [□ □ ■] down on any click, you can extend the selection based on the original selection mode, (character, word, or line). Extending the selection with [□ □ ■] will also be based on the original selection mode.

And remember — nothing happens until you execute a command. If you select the wrong amount of text, nothing will happen. Just start the selection over.

2.2.3 Click Method for Selecting Text

The click method is useful for selecting large areas of text.

Step 1: Click [■ □ □] at the beginning of the desired text selection.

Step 2: Move to the end of the desired selection. (You can scroll or use any other window moving commands to do this.)

Step 3: Click [□ □ ■].
The text between the [■ □ □] and [□ □ ■] click is now highlighted.

[■ □ □] starts the selection. [□ □ ■] extends (or contracts) the selections. You can also extend the selection by holding down the [□ □ ■] and moving the cursor.

2.2.4 Hold Down Button Method for Selecting Text

This method is convenient for selecting small portions of text.

Step 1: With the cursor at the front of the text to be selected, press and hold [■ □ □].

Step 2: Still holding [■ □ □], move the mouse cursor to the end of the desired selection. The selection will follow the mouse cursor.

Step 3: Release [■ □ □] when your selection is highlighted. Remember, use [□ □ ■] to extend or shorten your selection.

Now that you know how to select text, the following sections explain how to manipulate the text you have selected.

2.3 Move and Copy Text

Moving and copying text are different: text that you move is deleted from its original location and inserted to a new one; text that you copy stays in its original location and is duplicated at a new location.

There are several ways to move and copy text. The simplest way is with Del and Ins.

2.3.1 Move or Copy Using "The Scrap"

Del and Ins are useful for moving and copying text when you want to move small portions, or make several quick copies of text, but impractical for moving or copying large portions of text. Move and Copy are better for moving and copying large portions of text.

Step 1: Highlight the text you wish to move or copy.

Step 2: Press Del or F1. This deletes the selection to the scrap buffer. The scrap buffer holds the last section of deleted text until you are ready to insert it into another area of your text, or to replace it with another delete action.

Step 3: To keep text in its original location, re-insert it immediately with the Ins key.

Step 4: Move the mouse cursor to the place where you want to copy the text and click the ■ □ □ button. You are about to insert text *in front of the cursor.*

Step 5: Press Ins or F2. This inserts text from the scrap buffer to the insertion point.

2.3.1.1 Multiple Copies from the Scrap

Inserting text does not empty the scrap buffer; it merely copies its contents to the insertion point you choose. This means you can copy the same text as needed.

To make five copies of a line:

Step 1: Highlight the line. (triple click ■ □ □ on the line)

Step 2: Delete the line: press Del once.

Step 3: Insert it five times: press Ins five times.

2.3.2 Move or Copy Without the Scrap

Text that you *move* is deleted from its original location and inserted to a new one; text that you *copy* remains in its original location and is duplicated to a new location.

There are two ways to *move* text:

- Ctrl-■ □ □ ; and
- Extract F5 .

There are two ways to *copy* text:

- ⇑Shift-■ □ □ ; and
- Duplicate F4 .

2.3.2.1 Move Text, the Ctrl-Mouse Method

Step 1: Highlight the text to be moved.

Step 2: Move mouse cursor to the point where you want to move the text.

Step 3: Press Ctrl and click the ■ □ □ mouse button. When you release the ■ □ □ mouse button, the text you chose in **Step 1** will be inserted (moved) just in front of the mouse cursor to the insertion point.

2.3.2.2 Move Text, the Extract Method

Extract is used for moving text to a pre-selected insertion point.

Step 1: Highlight the point at which you want the text inserted.

Step 2: Press F5 .

Step 3: Now find the text you want moved and highlight it.

Step 4: Press F5 again. The text you just chose in **Step 3** is automatically moved to the insertion point you chose in **Step 1**.

2.3.2.3 Copy Text, the Shift Mouse Method

Copy requires that you select the text before you select the insertion point. Then (after you select the text to be copied and before you copy it) scroll or thumb the window to where the text will be copied. (The text to be copied does not need to be visible in a window at the time of the copy.)

Step 1: Highlight the text to be copied.

Step 2: Move the mouse cursor to the point where you want to copy the text.

Step 3: Press [⇑Shift] and click the [■ □ □] button. The text will be copied to the place just in front of the mouse cursor.

2.3.2.4 Copy Text, the Duplicate method

Duplicate lets you copy text to a preselected insertion point.

This method lets you build lines from pieces of other lines without disturbing the borrowed text.

Also, you do not lose the insertion point when selecting text.

Step 1: Select an insertion point in the new window with a [■ □ □] click.

Step 2: Press [F4].

Step 3: Highlight the text to be copied.

Step 4: Press [F4] to copy highlighted text to the insertion point.

Step 5: To copy additional text, repeat **Steps 2 through 4**.

NOTE

This concludes the step-by-step portion of the tutorial. The remainder of this tutorial explains the editing commands invoked from the `EDITING`, `MOVING` and `QUIT+ETC` menus.

2.4 The EDITING Menu

2.4.1 Redo Last Edit

Choose **`Redo Last Edit`** from the **`EDITING`** menu. It offers a quick way to repeat the last edit action performed.

The effect of **`Redo`** depends on the last editing action:

Copy Text copied in the last edit is copied to the selected insertion point.

Move Text moved in the last edit is moved to the selected insertion point.

Insert Text most recently typed is inserted at the selected insertion point.

Delete/Insert Text is deleted and replaced with previously inserted text.

Delete Only possible when followed by an insert.

Select a word, delete it, and type a different word. Then select a second word and choose **`Redo Last Edit`** from the **`EDITING`** menu, or [F8] .

Notice: the second word is replaced with the same text as the first word. **`Redo`** repeats [Del] [Ins] as one action.

2.4.2 Undo Last Edit

To undo the last edit, choose `Undo Last Edit` from the `EDITING` menu, or press F9.

Practice undoing some edits: Delete text, undo it; insert text, undo it; select a word, delete it and type in a different word. Now undo that.

> NOTE
>
> Del-Ins is defined as one action, and is undone as one action.

2.4.2.1 Undo (reversible) the last edit.

The last edit is undone, that is, its effect is nullified. The undo undoes the previous undo. This lets you switch between two versions and compare . You can undo any number of times switching between the two versions.

The effect of an undo on the various edits:

Copy	Copied text is deleted.
Move	Text that was moved is moved back.
Delete	Deleted text is reinserted.
Insert	Inserted text is deleted.
Delete/Insert	Inserted text is deleted and the deleted text is reinserted.

This undo is considered an edit, so a second undo undoes the effect of this undo. Undo can temporarily toggle between two versions of the same text.

2.4.2.2 Undo Multiple Edits

`Undo Multiple` can undo up to 50 previous edits.

For this, use `Undo Multiple` from the `EDITING` menu or ⇑Shift-F9.

Be careful here: this undoes edits *so that the undo cannot be undone.* Thus, a second `Undo Multiple` undoes an additional previous edit.

2.4.3 Global Replace

Replace is related to *Search*. It is called from the EDITING menu or with Alt-R. You are asked for the string to search for. Then you are asked for the string to replace it. Next, you are asked whether to perform the replacement from the selection to the end of the file (globally), or only within the selection. Finally, are asked if you want to verify each replacement. If you press Y (yes), you are asked each time the string is found whether you want it replaced.

Global Replace and Search are affected by the ignoreCase and searchMode options.

2.4.4 Insert ASCII Character

Insert ASCII lets you enter any ASCII character into your file. You must specify the character by its numeric value.

Type in the number of the character where desired in your text. Then highlight the number and select **Insert ASCII**. The ASCII character you defined will take the place of the number you typed.

Specify the numeric value in decimal, octal, or hexadecimal form:

Hexadecimal If the first two digits of the number are **0X**,
then the rest of the number is hexadecimal.

Octal If the first digit is **0** and the second is *not* **X**,
then the rest of the number is octal.

Decimal If the first digit is *not* **0**,
then the number is decimal.

Refer to your ***DOS*** or ***BASIC*** manual for further information on ASCII codes.

2.5 The MOVING Menu

2.5.1 Search for Text

The *POINT Editor* lets you search for words, phrases and other strings quickly, conveniently, and accurately.

When you invoke a search you are prompted for the string to search for. Type it in and end the string with the [↵] key. If the string is found, the window is moved (if necessary) and the string that was found is selected.

A string search starts at the current selection and goes to the end of the file. Invoke the **Search** command with `find` on the top line menu, with `Search for String` from the `MOVING` menu, or with [⇧Shift]-[F6].

When the *POINT Editor* requests a search string it automatically enters the last string you searched for as the default string. [↵] accepts the default string, or you can modify it with [← Back]. If you type any other character, the default string is erased and the string you type is used.

Search for a string; accept the default string. Try it again; this time type a new string.

You can also accept the default search string with the mouse. In general, [■ □ □] acts like [↵] and can be used to accept the default string. [□ □ ■] then causes the action to be performed on the string once it is found.

2.5.2 Crossing Windows

You can also select a string in one window and search for it in another. Here's how:

Step 1: Highlight the string you wish to search for.

Step 2: Make the window active where you want to search.

Step 3: Select `Search for String` from the `MOVING` menu.

If the selection is in the window, then the search starts at the selection. Otherwise, the search starts at the beginning of the file.

To search for the next occurrence of the selection, use [■ □ □] to select `next` from the main menu, `Search for Selection` from the `MOVING` menu, or press [F6].

2.5.3 Search Backwards

Searches backwards (towards the beginning of the file) for the selected string.

`Search Backwards` is invoked from the `prev` command on the main menu, or from the `MOVING` menu, or with Alt-F6, or with Ctrl-F6.

2.5.4 Search Options

Several search options are offered by *POINT*. Refer to Section 6.4 for detailed descriptions.

2.6 The QUIT+ETC Menu

2.6.1 Variations on the Quit Command

There are three ways to quit that determine what is done with the your edited files. Each quit can be called from the `Quit+Etc` menu or from the keyboard.

`Quit and Ask`

[Ctrl]-[F3], [Alt]-[F3], or [Alt]-[Q]

You are asked about each unsaved file. For each file, you must decide whether to save the edited version, or to discard it and keep the original. If you press [Esc] in response to any question, `Quit` is abandoned and you return to the *POINT* editing session.

`Quit-Save files`

[F3]

This automatically saves all edited and unsaved files and then exits to *DOS*.

`Quit and Discard Edits`

[⇑Shift]-[F3]

This exits to *DOS* without saving any files, unless there are any recently edited but unsaved files. It asks you to verify these.

2.6.2 More Commands from QUIT+ETC

2.6.2.1 Escape to DOS Shell

Select `Escape to DOS Shell` from the `QUIT&ETC` menu, or press Alt-D. This gives you a *DOS* prompt to execute any *DOS* command. Then, to return to your *POINT* session, use the *DOS* **EXIT** command.

2.6.2.2 Execute Selected DOS Command

Alternatively, you can type a *DOS* command in a window, highlight it, and choose `Execute Selected Command` on the `QUIT&ETC` menu. The selection is taken as a *DOS* command and passed to the *DOS* interpreter for execution. The highlighted command is executed and the standard output of the command is sent to a window (which is created by the *POINT Editor* for this purpose). The window appears when the command is completed.

The file name of the window is of the form **`UnNamed.x`**, where **`x`** is a letter in the range from **`a`** to **`z`**.

2.6.2.3 Get Help

This is an entry to the help system. From all help screens:

Spacebar	Returns to editing.
Esc	Returns to editing.
PgDn	Moves to the next screen. If there is no logical next help screen, you are returned to the general help menu.
PgUp	Moves to previous help screen.

2.6.2.4 Redraw the Screen

The entire screen is redrawn.

Chapter 3
POINT Concepts

Chapters 3 through 6 provide a complete reference to the ***POINT Text Editor*** and describes many features not mentioned in the tutorial, Chapters 1 and 2.

3.1 What POINT Is

The ***POINT Editor*** is mouse-based, with overlapping windows and user definable menus, and is designed to run on *IBM PC* and compatible computers. It can manage up to **20** windows simultaneously and can be easily configured to your specific needs.

3.2 How POINT Works

3.2.1 How POINT Works with a Mouse

3.2.1.1 Mouse Cursor

Since the ***POINT Editor*** only runs in *text mode*, the mouse cursor is a small block on the screen. It is controlled by your mouse driver, not by the ***POINT Editor***.

3.2.1.2 Mouse Buttons

The ***POINT Editor*** can be used by either a two-button mouse or a three-button mouse. Two-button commands are done by pressing [■ □], [□ ■], or [■ ■]. [□ ■ □] on a three-button mouse usually has the same effect as [■ ■] on a two-button mouse.

[■ □ □] is the main button. Use [■ □ □] unless specifically directed to use [□ □ ■].

Sometimes you hold a button while moving the mouse. Other times you use the keyboard [⇑Shift] or [Ctrl] with the mouse buttons. When you use [⇑Shift], [Ctrl], or [Alt] while clicking a mouse button, we recommend that you press and hold the key *before* you press the required mouse button.

It's not necessary to press and release [■ □ ■] simultaneously. Since action takes place when buttons are released, the ***POINT Editor*** monitors the buttons while they are down. When a button is down, if another button is pressed at the same time, ***POINT*** considers that they were both pressed, and invokes the command for both buttons.

Practice this with "thumbing." Move the mouse cursor to the left window border. Then, while holding [■ □ □], press [□ □ ■]. While still holding [□ □ ■], release [■ □ □]. Finally, release [□ □ ■]. This is the thumbing [■ □ ■] command, not scrolling.

3.2.2 How POINT Works without a Mouse

If you work on more than one system, you will occasionally work without a mouse. For this reason, a non-mouse interface is included here. Although non-mouse use is only occasional, we have made it as close as possible to the mouse interface.

Cursor keys move the mouse cursor, and other keys simulate mouse buttons.

Mouse button keys simulate *a completed click* rather than a *press* or a *release* by itself: you can't hold down a simulated mouse button.

Some mouse commands are simulated by two keystrokes on the numeric keypad on the right side of the keyboard.

3.2.2.1 Mouse Movement Simulation

The four cursor direction keys move the mouse cursor one row or one character. Ctrl-← or Ctrl-→ moves *one word at a time* to the left or the right. "*Word*" here means *a sequence of visible characters between two blanks.*

Fast cursor motion requires two keystrokes. The first keystroke is done with the Gray +, −, or Home on the numeric keypad.

Starting with the Gray +:

← and → keys move the cursor 10 columns *left or right.*

↑ and ↓ keys move the cursor 6 rows *up or down.*

Home, PgUp, End, and PgDn move the cursor *diagonally* 10 columns and six rows.

Starting with the Gray −:

A cursor arrow key moves the cursor to the window edge.

PgUp or PgDn moves to the top or bottom window border.

End moves the cursor to the right window border.

Home moves the cursor to the window scroll border.

Starting with Home:

A cursor arrow key moves to one of the four screen edges.

Home or End moves to the beginning or end of the current line.

Home, then ↑ moves to the top menu line.

These commands move **the mouse cursor only**. They do not affect text selection.

3.2.2.2 Mouse Button Simulation

Mouse button simulation also requires two keystrokes. The first keystroke is done with the End key on the numeric keypad.

Starting with the End key:

End	simulates	■ □ □
↓	simulates	□ ■ □
PgDn	simulates	□ □ ■
←	simulates	⇑Shift-■ □ □
Gray +	simulates	⇑Shift-□ ■ □
→	simulates	⇑Shift-□ □ ■
Home	simulates	Ctrl-■ □ □
↑	simulates	Ctrl-□ ■ □
PgUp	simulates	Ctrl-□ □ ■
Gray −	simulates	Alt-■ □ □

3.2.2.3 Using Mouse Keys with a Mouse

Mouse simulation keys work differently if you use a mouse.

Cursor movement keys move the mouse cursor and the selection. This is normally what you want when you have a mouse since it saves additional keystrokes to move the selection after the mouse cursor is moved. Since the selection moves, the screen must be redrawn after each cursor movement command. This makes these commands somewhat slower.

If the `cursorMouse` option is set to `1`, mouse cursor simulation will only move the mouse cursor even if a mouse is present.

3.2.2.4 Mouse Simulation Command Table

Key	Moves Mouse Cursor...
←	1 character left
→	1 character right
Ctrl-←	1 word left
Ctrl-→	1 word right
↑	1 row up
↓	1 row down

First Key	Next Key	Cursor Moves...
Gray +		
	←	10 columns left
	→	10 columns right
	↑	6 rows up
	↓	6 rows down
	PgUp	6 rows up and 10 columns right
	PgDn	6 rows down and 10 columns right
	Home	6 rows up and 10 columns left
	End	6 rows down and 10 columns left

First Key	Next Key	Next key moves *within window* to...
Gray −		
	←	Left edge of window
	→	Right edge
	↑	Top edge
	↓	Bottom edge
	PgUp	Top border
	PgDn	Bottom border
	Home	Left border
	End	Right border

First Key	**Next Key**	**Next key moves *within screen* to...**
Home		
	←	Left edge of screen
	→	Right edge
	↑	Top edge
	↓	Bottom edge
		Next key moves *within window* to...
	PgUp	Middle of window
	PgDn	Middle of window
		Next key moves *on the line* to...
	Home	Beginning of line
	End	End of line

First Key	**Next Key**	**Next key simulates mouse click...**
End		
	End	■□□
	↓	□■□
	PgDn	□□■
	←	⇑Shift-■□□
	Gray +	⇑Shift-□■□
	→	⇑Shift-□□■
	Home	Ctrl-■□□
	↑	Ctrl-□■□
	PgUp	Ctrl-□□■
	Gray −	Alt-■□□

3.2.2.5 Mouse Sensitive Points

There are several *mouse-sensitive points* on the screen where the mouse can be used to directly: move, stretch, or contract a window; start and extend the text selection; make a window the active window; top or bottom a window; and scroll the window vertically or horizontally.

In the following table, the left column shows the command name, the middle column shows the location of the mouse cursor, and the right column shows the mouse button to press. □ ■ □ means ■ ■ on a two-button mouse.

Command	Cursor Position	Mouse Button
--Not Defined--	Outside window	■ □ □
--Not Defined--	Outside window	□ □ ■
Display TOPLIST menu	Outside window	□ ■ □
Select text (characters)	Inside window	■ □ □
Extend selection	Inside window	□ □ ■
Display TOPLIST menu	Inside window	□ ■ □
Scroll up	Left border	■ □ □
Scroll down	Left border	□ □ ■
Thumb vertically	Left border	□ ■ □
Scroll left	Bottom border	■ □ □
Scroll right	Bottom border	□ □ ■
Thumb horizontally	Bottom border	□ ■ □
Make active window	Top border	■ □ □
Move window	Top border	□ □ ■
Split vertically	Top border	□ ■ □
--Not Defined--	Right border	■ □ □
--Not Defined--	Right border	□ □ ■
Split horizontally	Right border	□ ■ □
Top/Bottom window	Any corner	■ □ □
Stretch window	Any corner	□ □ ■
--Not Defined--	Any corner	□ ■ □

Chapter 4
Initialization Files

4.1 What They Do

Initialization files use the .INI extension are to be called in the current session. They also define the menus you see on the screen.

You can change top line commands and menus. You can also have several *different* top lines. You can even move the top line to the bottom of the screen.

By customizing the .INI files, you can also specify pop-up mouse menus available inside any window. A [?]-[❑ ❑ ❑] combination determines which menu is invoked. You can also specify the names and commands on pop-up mouse menus.

You can specify which mouse buttons invoke the select, extend, copy, and move functions. You can also specify options, and bind commands to keyboard keys.

4.2 How They Work

When *POINT* loads, it looks for the PT.INI file and incorporates the command and menu definitions defined there. INI files are user-definable.

After *POINT* finds and processes PT.INI, it checks the current directory for an additional file named PTLOCAL.INI. If it finds this file it processes it also. Since PTLOCAL.INI is processed after PT.INI, it can override the commands in PT.INI.

The global PT.INI sets the options, menus, keys, etc. the way you usually want them. PTLOCAL.INI can further change the way *POINT* behaves in a given directory.

4.2.1 The PT.INI File

PT.INI

The primary initialization file. Contains menu definitions, options and key bindings.

4.2.2 PTLOCAL.INI Files

PTLOCAL.INI

An initialization file with functions you can define to work in a specific sub-directory.

4.2.3 Other INI files

COLOR.INI

The color file. If you have a color monitor, copy COLOR.INI to PT.INI.

MONO.INI

The monochrome file, this file as well as the original PT.INI is set up for monochrome monitors (also for black and white monitors running with the color card).

If you have been using a color monitor and must now use a monochrome monitor, copy this file back into PT.INI. This file is a replica of the original PT.INI file.

PTEXPERT.INI

The expert file. Copy this file into PT.INI when you are more familiar with the *POINT Editor*.

PTM2.INI

Integrates the *POINT* editor with the programming environment for ***LOGITECH Modula-2, Version 3.0***.

4.3 Options

The following options can be included as is in your PT.INI file.

For these options the default is always listed first. The capitalization of the option names is done for clarity in processing the PT.INI file capitalization is ignored.

`43lines=0`

The screen shows the usual 25 lines of text.

`43lines=1`

You have the ***Enhanced Graphics Adapter*** and a monochrome display or the ***Enhanced Graphics Display.*** If `43lines=1`, *POINT* uses the smaller (8x8) character set which allows 43 lines on the screen (still with 80 characters per line).

`autoIndent=1`

When a newline (↵ key) is inserted, also insert the spaces and tabs that begin the previous line.

`autoIndent=0`

No special action on inserting a newline.

`buffers=50`

The number of 1024-byte memory buffers are allocated to hold parts of the files being edited. As you specify more buffers, *POINT* will work faster for larger files but it will also use up more space in memory. The value of "buffers" must be in the range of 2 to 300.

`cursorMouse=0`

When the mouse driver is present in memory, the cursor motion keys will actually move the selection. This is usually what you want but it is slower and does not really simulate mouse cursor motion.

`cursorMouse=1`

The cursor keys will only move the mouse cursor and will have no effect on the selection. This is much faster and provides complete mouse simulation but it means that you need an extra keystroke after cursor motion to make the selection. This is the default if the mouse driver is not present in memory.

`dosExit=C:\COMMAND.COM`

This controls which command processor is invoked when you execute the `Execute DOS Command Processor` command. Actually the default is the value of the COMSPEC variable. If you use some other command processor you can have that be invoked instead of COMMAND.COM.

`fullNames=0`

Only the filename is shown on the window banner and the default Toplist Menu.

`fullNames=1`

The full pathname of the file is shown.

`filePattern=*.*`

This is a DOS file name pattern with the wildcard character '*' and '?' allowed. It is used by ***POINT*** to determine which file names are displayed for mouse selection when you load a file into a window. The pattern can contain a **PATH** name. For example, `filePattern;=\wp\*.doc` is a valid `filePattern`.

You can specify several file patterns separated with a vertical bar (|). ***POINT*** will display all matching file names. For example, `filePattern=*.txt|\etc\*.dat|*.prg` is a valid `filePattern`.

`helpMode=2`

Whenever a top line or menu command is touched, a short, one-line description of the command is written on the bottom line of the display.

`helpMode=1`

Whenever a menu command is touched, a short, one-line description of the command is written on the bottom line of the display.

`helpMode=0`

No description messages are displayed.

`ignoreCase=0`

Search and Replace commands — consider case in the string being searched. No match is made unless the characters and the cases both are equal.

`ignoreCase=1`

Search and Replace commands — ignore case in the string searched.

`ignoreCase=2`

Searches strings with upper case letters are case sensitive; others are insensitive. Applies to `Search for String`, not to `Search for Selection`.

`initialWindows=0`

Each file listed on the command line is put in a full screen window and the windows overlay each other.

`initialWindows=1`

The first window takes up the whole screen, the rest of the windows are evenly spaced horizontally down the screen and are all the whole width of the screen.

`initialWindows=2`

The first window takes up the whole screen. The rest of the windows are evenly spaced vertically across the screen and are all the whole length of the screen.

`linesOverFind=0..4`

`linesOverFind=0` determines placement in the window of a search string when it is found by a search command. If the window must be scrolled vertically, `linesOverFind` determines how many lines from top of the screen the found string will be placed. If the value of `linesOverFind` would place the string below the bottom of the window, the search string is placed in the middle of he window. The default value is `linesOverFind=4`.

`makeBaks=1`

Make .BAK backup files when a file is edited and saved.

`makeBaks=0`

Do not make .BAK files.

`noRecurse=0`

Do not complain if a second copy of *POINT* is started.

`noRecurse=1`

If *POINT* finds an existing *POINT* work file in the current directory, it assumes that a copy of *POINT* is already running and issues a warning message to that effect. This warning message is often caused by an old *POINT* work file that was not deleted because the system was turned off while *POINT* was running.

`overType=0`

Insert characters that are typed.

`overType=1`

Typed characters replace screen characters, except for the end-of-line character.

`readOnly=0`

Files **can be** written back to disk using the same file name.

`readOnly=1`

Files **cannot be** written back to disk using the same file name.

rightBack=0

[■ □ □] scrolls up, and [□ □ ■] scrolls down.
This has no effect on the meaning of [PgUp] and [PgDn] .

rightBack=1

[□ □ ■] scrolls up (backwards in the file) and [■ □ □] scrolls down.

rightBack=2

[■ □ □] scrolls up, and the [□ □ ■] scrolls down. Scrolling up with the mouse causes the line the mouse cursor is on to become the bottom line.

rightBack=3

[□ □ ■] scrolls up (backwards in the file) and [■ □ □] scrolls down. Scrolling up with [□ □ ■] sends the mouse cursor line on to the bottom of the screen.

rightMargin=999

The right margin controls the word wrap feature of ***POINT***. When you are typing in text and you pass the right margin, the word you are typing to moved to the next line automatically and a newline is inserted in front of it. A "word" is a sequence of characters delimited by blanks or tabs.

searchMode=0

String searches start at the selection (or the beginning of the file if the selection is not in the window) and go to the end of the file.

searchMode=1

Search from the selection backwards to the beginning of the file.

searchMode=2

Start search from the selection, go to the end of file, continue at the beginning of file, and go to the selection. **searchMode=2** *always succeeds*, since it finds the original selection (after searching the whole file) even if the selection does not occur any other place in the file.

searchMode=3

Search for string in all the windows. Start at the active window and search the files in each window below it until the bottom window. ***POINT*** message issues telling you which file is being searched (although they go by more quickly than you can read them, unless you have very large or heavily edited files).

When the string is found, all the windows on top of the window in which the string is found are *bottomed*. Another search will continue in this and the following windows.

`tabWidth=8`

The tab stops are set every 8 places. Any value can be set.

`tiledSplit=0`

When a window is split it stays the same size, and the split window is half its size.

`tiledSplit=1`

When a window is split it is reduced to half its size, and the split window takes up the other half of the screen space it was using.

`topOnFind=0`

When a string is found in a window, the window is NOT automatically made the top window. This means that you may not be able to see the string that was found since it might be beneath another window.

`topOnFind=1`

When a string is found in a window, the window is automatically made the top window.

`undoBack=0`

`Undo Last Edit` records its action in the change history and can only undo one previous action.

`undoBack=1`

`Undo Last Edit` acts like **`Undo, Erase History`**.

`undoSize=50`

How many previous changes are remembered that can be undone. The value of **`undoSize`** must be in the range **2** to **100**.

`unixMode=0`

Use *MS DOS* line ending convention for all files. This is the default value.

`unixMode=1`

Use *UNIX* line ending convention for all files.

`unixMode=2`

Use the convention that appears to be correct for each file.

The *POINT Editor* examines the first 300 characters of each file as it is read. If any of these characters is a carriage return (decimal 15) it uses the *DOS* line ending convention for that file, otherwise it uses the *UNIX* convention. This decision is made *once* for a file and not changed until another file (or the same file) is loaded into the window.

`videoMode=0`

For monochrome or EGA adapters. No synchronizing is done on screen updating. This is the fastest method.

`videoMode=1` The display is written using the BIOS calls. This mode is the slowest but it should work in the widest range of environments and machines.

`videoMode=2`

The display is written (two bytes at a time) during the horizontal retrace. This mode should not cause any flicker (or snow) on a color graphics display but it is slower than **`videoMode=0`**.

`videoMode=3`

The display is written (one byte at a time) during the horizontal retrace. This mode will not cause any flicker (or snow) on a color graphics display but it is slower than **`videoMode=2`**.

`workDrive=.\`

The ***POINT*** work file will be put on this drive and directory. The default is the current drive and directory. You can use this to put the work drive on a RAM disk. The string can be a drive letter (and colon), a directory name, or a drive letter (and colon) and directory name. If you include a directory name, be sure to end it with a backslash (\ ).

4.3.1 Color Options

These options set foreground and background colors for various parts of the screen. For monochrome displays, they set video attributes for the characters:

- normal video,
- reverse video,
- underlined,
- blinking,
- intense.

Each option consists of several pairs of hexadecimal digits. In each pair the first digit is the background color and the second digit is the foreground color.

The color code is defined by *IBM PC* hardware. The hex color codes are:

Code	Color	Code	Color
0	Black	8	Grey
1	Blue	9	Light Blue
2	Green	A	Light Green
3	Cyan	B	Light Cyan
4	Red	C	Light Red
5	Magenta	D	Light Magenta
6	Brown	E	Yellow
7	White	F	Bright White

Only the first eight color codes (**0**-**7**) are supported for background colors. Using any of the other eight color codes (**8**-**F**) will cause the character to blink. It is possible to reprogram the display hardware so that color codes **8**-**F** cause bright backgrounds rather than blinking characters. You can do this reprogramming from the interactive color setting screen (described below).

`textColors=0770`

The first pair is the color of ordinary text and the second pair is the color of selected text. Each window can use different `textColors`.

`borderColors=70070F`

The first pair is the color of banner/menu line of the top of the window.

The second pair is the color of the left, right, and bottom borders.

The third pair is the color of the elevator on the left border. Each window can use different `borderColors`.

`msgColors=070FF070`

The first pair is the color of informational messages.

The second pair is the color of user input to the prompt, parts of the new/load file selection screen, and a few error messages.

The third pair is the color of error messages.

The fourth pair is the color of the top line containing commands and menus (if you have one).

4.4 Command Codes

Always specify commands in .INI files by their numbers.

4.4.1 Code and Associated Functions

Here are numbers and their associated *POINT* commands:

Number	Command
-1	No action.
0	No action.
1	Enter a character. (Do not use.)
2	Duplicate. Enter or exit duplicate mode.
3	Extract. Enter or exit extract mode.
4	Delete the selection to the scrap.
5	Quit and ask about edited files.
6	Set the value of the "debug" variable.
7	Create a new window.
8	Jump to the beginning of the file.
9	Jump to the end of the file.
10	Redraw the screen.
11	Zoom the window.
12	Top the window.
13	Change the size of the window.
14	Scroll up.
15	Scroll down.
16	Jump to a line number.
17	Close window.
18	Write the file to a new name.
19	Search for a string.
20	Global replace.
21	Display debugging information.
22	Cancel duplicate and extract modes.
23	Insert ASCII character.
24	Redo.
25	Undo (reversible).

Number	Command
26	Select text.
27	Search for the selection.
28	Top/Bottom the window.
29	Bottom the window.
30	Execute the selection.
31	Do nothing.
32	Execute the DOS command interpreter.
33	Load a new file into the window.
34	Move the cursor up.
35	Move the cursor down.
36	Move the cursor left.
37	Move the cursor right.
38	Hide the window.
39	Extend the selection.
40	Undo (multiple).
41	Simulate mouse buttons.
42	Fast cursor movements.
43	Jumps to the edges and borders of the window.
44	Jumps to the edges of the screen.
45	First Mouse Motion Command.
46	Second Mouse Motion Command.
47	Save all unsaved files.
48	Quit and save all edited files.
49	Quit and discard all edits.
50	Invoke user menu 1.
51	Invoke user menu 2.
52	Invoke user menu 3.
53	`TOPLIST` menu (the file in each window).
54	Save the file in the window.
55	Insert from scrap.
56	Go back to the last place you were.
57	Copy to scrap.
58	Show the selection.
59	Toggle insert/overtype mode.
60	Invoke user menu 4.

Number	Command
61	Delete, not to scrap.
62	Exchange the selection with the scrap.
63	OPTIONS menu.
64	Invoke the help system.
65	Invoke user menu 5.
66	Search in the reverse direction.
67	Invoke user menu 6.
68	Toggle 43-line mode (for EGA only).
69	Copy the selection to this spot.
70	Move the selection to this spot.
71	Change the menu (do not use).
72	Close and automatically save the file.
73	Load the selected file name into active window.
74	Move cursor one word (blank delimited) left.
75	Move cursor one word (blank delimited) right.
76	Begin/end recording keystrokes in macro buffer.
77	Play back keystrokes from macro buffer.
78	Move Window and selection to the beginning of the file.
79	User-defined menu seven.
80	User-defined menu eight.
81	Change case (Upper/lower) of the first character of the selection.
82	Create a window and load the selected file name.
83	Find matching bracket: (or), [or], or { or }.
84	Go to the selected line number.
85	Justify all the lines selected between Left Border and Right Margin
86	Toggle Read-Only Status.
87	Change the Color Combination.
88	Exchange the two top windows.

4.4.2 Functions and Associated Code

	Number	Command
Window Management	7	New Window
	82	New Window From Selection
	38	Hide Window
	17	Close Window
	72	Close Window And Save File
	—	Split Window
	10	Redraw Screen
Position Window	12	Top Window
	29	Bottom Window
	28	Top/Bottom Window
	11	Zoom Window
	13	Change Window Size
	—	Stretch Window From A Corner
	—	Move Window
Position File in Window	15	Scroll Down
	14	Scroll Up
	—	Scroll Left
	—	Scroll Right
	—	Thumb Vertical
	—	Thumb Horizontal
	19	Search For String
	27	Search For Selected String
	66	Search Backwards For Selected String
	20	Replace String
	8	Go To Beginning Of File
	9	Go To End Of File
	16	Go To Line Number
	84	Go To Selected Line Number
	56	Go Back To Last Place
	58	Go To And Show Selection
	83	Finding Matching Bracket
	78	Beginning of the File and Move Selection

	Number	Command
File Commands	33	Load A New File Into Active Window
	73	Load Selected File Name Into Active Window
	54	Save File In Window
	46	Save All Unsaved Files
	18	Save File In Window Using Another Name
Select Text	26	Select Text
	39	Extend Selection
Insert Characters	—	Insert Character
	23	Insert ASCII Character
Copy And Move Text	69	Copy Text
	2	Duplicate Text
	70	Move Text
	3	Extract Text
	4	Delete Selection to the Scrap Buffer
	61	Delete Selection but Preserve the Scrap Buffer
	57	Copy Text To Scrap Buffer
	55	Insert Text From Scrap Buffer
	62	Exchange Selection with Scrap Buffer
	22	Cancel
	59	Toggle Insert/Overtype Mode
Redo And Undo Edits	24	Redo Last Edit
	25	Undo Last Edit (Reversible)
	40	Undo Edit (Multiple Step Undo)
Quit Commands	5	Quit And Ask About Unsaved Files
	48	Quit And Save All Unsaved Files
	49	Quit And Discard All Edits

	Number	Command
Execute DOS Commands	30	Execute Selection and put Output in a Window
	32	Escape to a DOS Command Interpreter
Macro Commands	76	Begin/End Recording Keystrokes
	77	Play Back Recorded Keystrokes
Miscellaneous Commands	6	Set Value of the "debug" Variable
	21	Display POINT Information
	64	Invoke Help System
	68	Toggle 43-line mode (EGA)
Invoke Menu Commands	50	Invoke User Menu 1
	51	Invoke User Menu 2
	52	Invoke User Menu 3
	60	Invoke User Menu 4
	65	Invoke User Menu 5
	67	Invoke User Menu 6
	79	Invoke User Menu 7
	80	Invoke User Menu 8
	53	Invoke TOPLIST Menu
	63	Invoke OPTIONS Menu
Keyboard Cursor Movement Commands	34	Move Cursor Up
	35	Move Cursor Down
	36	Move Cursor Left
	37	Move Cursor Right
	42	Fast Cursor Movement
	43	Move Cursor to the Edges of the Window
	44	Move Cursor to the Edges of the Screen
	41	Simulate Mouse Buttons
	74	Move Cursor One Word (Blank Delimited) Left
	75	Move Cursor One Word (Blank Delimited) Right

4.5 Menu Commands

All menus (top line, bottom line, and pop-up menus) are specified the same way. You can define up to eight menus. Menus are numbered 1 to 8 and their command numbers are:

Menu #	Command #
1	50
2	51
3	52
4	60
5	65
6	67
7	79
8	80

A menu specification starts with

[*an open bracket*
followed by the *menu number*

N (where N is a number in the range 1 – 8)
followed by one or more *menu items*, and

] *a close bracket*

A *menu item* is specified as a string in double quotes (" "), an equal sign (=), and a command number:

```
"<Command Name>"=<commandNumber>
```

"Command Name", or whatever title you choose is what you want to see in the menu on the screen. It can have spaces within the quotation marks, but not around the "=" or within <commandNumber>. It can use any of the 256 characters in the *IBM PC* extended character set.

<commandNumber> is the *POINT* command number as listed above.

A menu can contain up to 24 items (that's all that will fit on the screen). The "Command Name" can be as long or short as you want although there is a limit of 1400 characters for all names in all menus.

The first menu item is treated differently from the rest. The "command name" of the first item is taken to be the *title* of the menu. If you don't want a title, you can use an empty string (" "). Top line menus do not use titles, so their title should be empty. Remember, though — the first item is the *title* item, even for top line menus.

The command number of the title item is the command for the menu. This is the command that will be executed if you do not select any menu item. Normally this is called the null command. Use a command number of -1 or 0 for the null command.

See the .INI files on your ***POINT*** diskette for examples of menu specifications.

NOTE

Do not include angle brackets < or > or spaces in the string, or around the = sign. Strings like the above must be separated from other strings of the same type by one or more blanks, tabs, or new lines.

4.5.1 Top Line Menus

To specify a top line menu (as defined above), include a line in PT.INI with this format:

```
tXY=<commandNumber>
```

where "<commandNumber>" is (as before) a ***POINT*** command number. To specify a menu, use one of the menu command numbers: 50, 51, 52, 60, 65, or 67. You can use either an upper case "T" or a lower case "t".

"X" tells whether you want to alter the state of the ?-❑ ❑ ❑ that calls this menu. "X" is a digit from 0 - 7 which is a shift key code. The codes are:

X Code	Shift state specified
0	⇑Shift, Ctrl, and Alt keys are *all up*
1	⇑Shift down
2	Ctrl down
3	⇑Shift-Ctrl down
4	Alt down
5	⇑Shift-Alt down
6	Ctrl-Alt down
7	⇑Shift-Ctrl-Alt down

"`Y`" is the state of the mouse button keys you require to be down to invoke this menu. "`Y`" is also a digit from `0 - 7`.

Y Code	**Button state specified**
1	[■ □ □]
2	[□ □ ■]
3	[■ □ ■]
4	[□ ■ □]
5	[■ ■ □]
6	[□ ■ ■]
7	[■ ■ ■]

The menu is invoked if the [?]-[□ □ □] matches the "`XY`" you specify. To have several different combinations, include several `tXY=<commandNumber>` lines in PT.INI with the same `<commandNumber>`, one for each combination.

You can specify several different menus as top line menus. The one with the lowest hexadecimal "`XY`" number is the default menu, and is displayed first.

If you move the mouse cursor to the top line and press a button, the appropriate menu for that [?]-[□ □ □] will appear. No action takes place until you release the button. You can change buttons while on the top line to look at the menus available. If you move the mouse cursor from the top line before releasing the buttons, the default command of the menu will be executed. It is safest to make this a "do nothing" command. If you have only one top line menu, the menus and commands on it will be executed when you press the correct mouse button on the top line.

For example, the following line in PT.INI uses `menu 1` as the top line menu. (Command number 50 is `menu 1`).

```
    t01=50
```

If you specify a non-menu command number as a top line menu number, then that command will be executed when *POINT* detects the specified [?]-[□ □ □] at the top line of the screen. Thus, you can program certain commands to be invoked at the click of a mouse button on the top line. For example, to invoke `New Window` whenever you click [□ □ ■] on the top line, specify:

```
    t02=7
```

4.5.2 Pop-up Menus

You can also tell a menu to pop up whenever you press certain [?]-[□ □ □] combinations, either inside or outside a window. Specify a pop-up menu as follows:

```
bXY=commandNumber
```

where "`XY`" and "`commandNumber`" are as described in top line menus above. Top line menus are called when the correct [?]-[□ □ □] combination is seen while the mouse cursor is on the top line of the screen. These pop-up menus are invoked when the correct [?]-[□ □ □] combination is seen and the mouse cursor is inside any window or outside all windows.

For example, look at the following lines in PT.INI:

`b01=50 menu 1` (command #50) to pop up when you press [■ □ □]
`b02=51 menu 2` to pop up when you press [□ □ ■], and
`b03=52 menu 3` to pop up when you press [□ ■ □],
(or [■ ■] on a 2-button mouse).

4.5.3 Bottom Line Menus

You can also place your "top line" on the *bottom* of the screen, if you wish. Specify *bottom line menus* in exactly the same way as top line menus only use "`l`" or "`L`" instead of "`t`" or "`T`" as follows:

```
lXY=<commandNumber>
```

Everything else is the same as top line menus.

You cannot mix top line and bottom line menus. The menu that ***POINT*** sees first prevails. You will get error messages on menus that are not of the same type (either top or bottom), and they will be changed to the first type encountered.

4.6 Mouse Motion Commands

4.6.1 First Mouse Motion Command

The command number for mouse motion is **45.**
[□ ■ □] does this with the following line in PT.INI:

```
b04=45
```

If you use a two button mouse, attach mouse motion to [□ ■] with the line:

```
b02=45
```

If you attach the mouse motion command to [□ □ ■] , change the **NO MOTION** subcommand to extend so as not to lose the ability to extend the selection with the mouse. Then [□ □ ■] will act almost as it did before except you will be able to issue mouse commands with it also. What you will lose is the ability to drag the selection when extending it (that is, pressing [□ □ ■] and moving it to adjust the selection). It is not usual to do this anyway so the actual loss of function is small and the gain of having eight other mouse motion commands immediately available with [□ □ ■] is great.

4.6.2 First Mouse Motion Subcommands

Subcommands can be changed in PT.INI with lines of the form:

`mN=commandNumber`

where **`N`** is in a range from **`0`** to **`8`** as in the chart below:

```
              N[1]
   NW[8]       |     NE[2]
        \      |    /
         \     |   /
W[7] <----   [0]   ----> E[3]
           /   |   \
          /    |    \
   SW[6]       |     SE[4]
              S[5]
```

To change the no motion action to extend use:

`m0=39` — 39 is the extend selection command

Here are some other change ideas:

`m3=29` — Right motion bottoms the window
`m7=57` — Left motion copies the selection to the scrap

4.6.3 Second Mouse Motion Command

[Command #46]

Default subcommands of the second mouse motion command are:

```
             move to
           last place
   close       |      delete-
   window      |     no scrap
          \    |    /
           \   |   /
copy to <--- extend ---> exchange
 scrap      /  | \      with scrap
           /   |  \
          /    |   \
    bottom     |    cancel
    window     |
            TOPLIST
```

The command numbers are:

extend	#39
move to last place	#56
delete/no scrap	#61
exchange with scrap	#62
cancel	#22
TOPLIST	#53
bottom	#29
copy to scrap	#57
close window	#17

The second mouse motion commands can be changed in PT.INI with lines of the form:

```
nN=commandNumber
```

where N is 0, 1, 2, 3, 4, 5, 6, 7, or 8 as in the chart above.

4.6.4 Other Mouse Motion Parameters

You can also adjust *POINT* parameters to decide mouse motion direction.

Parameter **1** chooses between north and northeast. Parameter **3** chooses between northeast and east. We will talk about parameter **2** later. These parameters are the ratio of the number of rows you moved north and the number of columns you moved east multiplied by 100 so that they are integers. The default values are **175** and **25.**

Set these values in PT.INI. The default values are:

```
q1=175
q3=25
```

Experiment with these values if the defaults do not seem natural to you. These same parameters are used for the four other quadrants of the plane.

If you specify the diagonal commands to be **-1** or **31** (no action), *POINT* uses the nearest vertical or horizontal command. This can be used to reduce the possible motions to just four: **north, east, west, south** and **no motion**. Parameter **2** is used in this case to decode between north and east. The default value is **45** which is about a **45** degree angle. (That they are both **45** is a coincidence.)

4.7 Mouse Sensitive Window Points

You can decide (via lines in PT.INI) what command is executed when certain mouse buttons are clicked on parts of a window. Those you can change are listed below with the string used to set them in PT.INI enclosed in [square brackets].

[wtlm]	(□ ■ □) or (■ □ ■)	*top left corner*
[wtll]	(■ □ □)	*top left corner*
[wtrm]	(□ ■ □) or (■ □ ■)	*top right corner*
[wtrl]	(■ □ □)	*top right corner*
[wblm]	(□ ■ □) or (■ □ ■)	*bottom left corner*
[wbll]	(■ □ □)	*bottom left corner*
[wbrm]	(□ ■ □) or (■ □ ■)	*bottom right corner*
[wbrl]	(■ □ □)	*bottom right corner*
[wrbl]	(■ □ □)	*right border*
[wrbr]	(□ □ ■)	*right border*

Set these commands with lines in PT.INI of the form:

```
wtlm=88                 (exchange the top two windows)
wtrm=88
wblm=88
wbrm=88
wtll=28                 (top/bottom the window)
wtrl=28
wbll=28
wbrl=28
wrbl=27                 (search for selection)
wrbr=66                 (search backwards for the selection)
```

These lines also indicate default commands: (□ ■ □) on any corner swaps the two top windows, (■ □ □) on the right border searches up for the selection and (□ □ ■) on the right border searches down for the selection.

You may want to let (□ ■ □) on the top left corner be the *Beginning of File* and *Move Selection* command (see Section 3.2).

4.8 Mouse Button Commands

The method described in the previous section can be used to assign any command (not just a menu) to a [?]-[□ □ □] combination. This is most often used to assign the select, extend, copy, and move functions to mouse buttons inside windows.

For example, the lines below assign *select* to [■ □ □], *extend* to [□ □ ■], *copy* to [⇑Shift]-[■ □ □], and *move* to [Ctrl]-[■ □ □].

`b01=26`	[■ □ □]	*select*
`b02=39`	[□ □ ■]	*extend*
`b11=69`	[⇑Shift]-[■ □ □]	*copy selection to here*
`b21=70`	[Ctrl]-[■ □ □]	*move selection to here*

The .INI files on your disk contain examples of defined menus and functions assigned to top line menus and pop-up menus. They also suggest ways to effectively use these menu definition facilities. Start by using these .INI files, and modify them as you gain experience.

4.9 Key Commands

You can assign *POINT* commands to any key. However, the standard *DOS* keyboard handler does not let you reset certain key combinations (e.g., Ctrl-↑ and Ctrl-↓). In addition, certain keys are intercepted by *DOS* before the *POINT Editor* sees them.

Define keys with a line in PT.INI of the form:

```
kN=<commandNumber>
```

Either "`k`" or "`K`" tells *POINT* that a key is about to be defined, `N` specifies the key being defined, and `<commandNumber>` is a *POINT* command number as defined above. To define an *ASCII* code, let N be the *ASCII* character, specified as a *decimal number*. Use any *ASCII* chart (as in ***Appendix G*** of the ***IBM PC BASIC Manual***) to find the character codes. We include one here also.

To define one of the other keys on the PC keyboard (such as PgDn, Del, ←, etc.) start N with the zero (0) digit and then the extended code number of that key. We also list them here for your convenience. The initial "0" required in N is included so that the *POINT Editor* will know this is an extended key code and not an *ASCII* character.

A table of extended code numbers is in **Appendix G** of the ***IBM PC BASIC Manual.***

Let's look at some sample lines that might appear in PT.INI. These lines redefine three keys.

`k059=7`	F1	calls `New Window`
`k14=7`	Ctrl-N	calls `New Window`
`k082=59`	Ins	toggles between overtype and insert modes

The following tables shows command functions that can be attached to keys as defined in the PT.INI file.

The format for this definition is

```
k<N>=<Command Number>
```

where

`k` or `K` indicates that a key is about to be defined,
`<N>` is one of the key codes defined in Section B.1, and
`<Command Number>` is a number defined in Section B.2.

ASCII Code	Character	ASCII Code	Character
0	Ctrl-2	32	Spacebar
1	Ctrl-A	33	!
2	Ctrl-B	34	"
3	Ctrl-C	35	#
4	Ctrl-D	36	$
5	Ctrl-E	37	%
6	Ctrl-F	38	&
6	Ctrl-F	39	’
7	Ctrl-G	40	(
8	← Back	41	)
9	Ctrl-I	42	*
10	Ctrl-↵	43	+
11	Ctrl-K	44	,
12	Ctrl-L	45	-
13	↵ or	46	.
	Ctrl-M		
14	Ctrl-N	47	/
15	Ctrl-O	48	0
16	Ctrl-P	49	1
17	Ctrl-Q	50	2
18	Ctrl-R	51	3
19	Ctrl-S	52	4
20	Ctrl-T	53	5
21	Ctrl-U	54	6
22	Ctrl-V	55	7
23	Ctrl-W	56	8
24	Ctrl-X	57	9
25	Ctrl-Y	58	:
26	Ctrl-Z	59	;
27	Esc or	60	<
	Ctrl-[	61	=
28	Ctrl-\	62	>
29	Ctrl-]	63	?
30	Ctrl-6	64	@
31	Ctrl--		

ASCII Code	Character	ASCII Code	Character
65	A	97	a
66	B	98	b
67	C	99	c
68	D	100	d
69	E	101	e
70	F	102	f
71	G	103	g
72	H	104	h
73	I	105	i
74	J	106	j
75	K	107	k
76	L	108	l
77	M	109	m
78	N	110	n
79	O	111	o
80	P	112	p
81	Q	113	q
82	R	114	r
83	S	115	s
84	T	116	t
85	U	117	u
86	V	118	v
87	W	119	w
88	X	120	x
89	Y	121	y
90	Z	122	z
91	[	123	{
92	\	124	|
93	]	125	}
94	^	126	~
95	_	127	Ctrl-← Back
96	`		

Extended Code	Characters
003	NUL
015	⇑Shift-→|
016-025	Alt plus Q W E R T Y U I O P
030-038	Alt plus A S D F G H J K L
044-050	Alt plus Z X C V B N M
059-068	F1 through F10 (unshifted)
071	Home
072	↑
073	PgUp
075	←
077	→
079	End
080	↓
081	PgDn
082	Ins
083	Del
084 - 093	⇑Shift-F1 through ⇑Shift-F10
094 - 0103	Ctrl-F1 through Ctrl-F10
0104 - 0113	Alt-F1 through Alt-F10
0114	Ctrl-PrtSc
0115	Ctrl-←
0116	Ctrl-→
0117	Ctrl-End
0118	Ctrl-PgDn
0119	Ctrl-Home
0120 - 0131	Alt plus 1 2 3 4 5 6 7 8 9 0 - =
0132	Ctrl-PgUp

Chapter 5
How POINT is Displayed

This chapter describes how text appears, screen layout, how windows appear, how text is handled in the windows, how files are managed, and a general overview of ***POINT*** commands.

Remember: there are three primary functions on your ***POINT*** screen:

Selection cursor

A **blinking square** that can be moved by the mouse — or, if you have no mouse, by keypad cursor keys — to either select a new insertion point, or so select a command outside the text window. The new selection or command is then implemented with [■ □ □] (and sometimes also with [□ □ ■]).

Selection

An **area of text that is highlighted** for purposes of manipulation, or for indicating the insertion point.

Insertion point

The **area at the beginning of the selection** at which new characters appear from your keyboard, or as the result of a move or copy operation from another area of text (from the same window, or even from a different file in another window).

One last word here: the insertion point is set either to **add characters** at the insertion point (`OverType=0`) or to replace existing text with new text (`OverType=1`).

5.1 The POINT Screen on your Monitor

5.1.1 The Main Screen

5.1.1.1 The Top Line

The top line of the screen contains command names and selected menus.

To select a top line command, click the (■ □ □) on the command name.

To select a menu option, move the mouse cursor to the menu line and press and hold the (■ □ □) on a menu name. This pulls down the menu. Then *drag* the highlighted cursor into the menu and release the button on the desired option.

5.1.1.2 The Working Area

The main body of the screen is the work area, where files are loaded and edited.

5.1.1.3 The Bottom Line

The bottom line of the screen initially displays the version and copyright notice. When editing begins and a menu command in invoked the bottom line becomes a comment line. The **comment line**, which alternates between simple help reminders and command prompts, is activated when the cursor is moved to the **menu line**. Move the blinking cursor to the menu line. Notice how the copyright notice temporarily becomes a comment.

5.1.2 The Selection Screen

When you select `open` or `New Window` from the top line of the main screen, a selection screen displays the file names in the current working directory.

5.1.2.1 Top Area

There are three lines at the top area of the selection screen. *Line one* displays the directory `Patterns` the files are sorted by and the `Number of files` in the current directory. *Line two* displays command options you may invoke with the cursor. *Line three* displays user information.

5.1.2.2 Work Area

The *work area* displays a list of file names in the current directory.

5.1.2.3 Bottom Line

The *bottom line* prompts for a file name to load the new window.

5.1.3 The Color Option Screen

The color option screen is invoked from the `Options` menu by selecting one of three color options. The color option screen is a simple way to modify the color scheme of the active window. If you choose `colors` from the `OPTIONS` menu, you are shown a screen consisting of three major parts.

The Top Area is a menu line with 10 commands. Nine of these are used to select the part of the display that you want to change the color of. The part that will be changed is selected by clicking on its name on this menu line. The tenth command exits interactive color setting.

The Work Area is a line with four commands and a menu of the 128 possible combinations of foreground and background colors. Select a color combination by clicking on it. These commands let you look at another 128 color combinations (the alternate colors) which will either be blinking or have high-intensity background colors. Two commands switch you between plain and alternate color sets. The other two commands alternate the color set from regular to blinking to intense.

The Bottom Area is a sample window that shows you the effect of the changes you have just made on the color options screen.

5.2 The Window on your POINT Screen

5.2.1 The Banner Line

The banner line at the top of each window displays:

- SAVE when editing space is low.
- *Name of the file* loaded in the window.
- * if the file has been *changed.*
- *Line numbers* shown in the window.
- *Column numbers* shown in the window.
- OverType when *POINT Editor* is in *overtype mode.*
- *Unused space* filled with blanks or double bars.

5.2.1.1 The SAVE Message

The *POINT Editor* uses an "edit space" to keep track of changes: the space available here for editing depends not on *file size*, but rather on the *number of changes* you make in the files.

When editing space is low, SAVE appears in front of the file name in each window. When this happens, save one or more edited files to reclaim the editing space.

5.2.1.2 The File Name

The file name displayed is exactly as you typed it (including capitalization) and may include a **PATH** name. If the file in the window has been changed, a * (asterisk) appears after the file name.

5.2.1.3 Line and Column Numbers

Line numbers and column numbers are given as a range. Column numbers can be useful when scrolling horizontally.

5.2.1.4 The OverType Message

`OverType` shows in the banner line if keystrokes replace existing text.

5.2.1.5 The Banner Line Fill Character

The unused part of the banner line *is* blank, unless it is the active window. In the active window the double line goes through the unused part of the banner line.

5.2.1.6 Banner Line Mouse Commands

If you press [□ □ ■] on the window banner line, a shadow border follows the mouse cursor and lets you move the window anywhere on the screen. The window gets smaller if you run into the edge of the screen, but grows back again as you move away from the edge.

[■ □ □] on a banner makes its window active.
[□ ■ □] on a banner splits the window.
[□ □ ■] on a banner can drag the window to a new area on the screen.

5.2.2 The Border

The active window has a *double line* border.
All other windows have a *single line* border.

[□ ■ □] on the right window border *splits* the window.

[■ □ □] [□ ■ □] or [□ □ ■] on the *bottom* window border scrolls thumbs *horizontally.*

[■ □ □] [□ ■ □] or [□ □ ■] on the *left* window border scrolls or thumbs *vertically.*

5.2.3 The Elevator

The left border of the window has an "elevator" that indicates both the position of the window in the file and the amount of the file shown in the window. The left border *between the two corners* represents the whole document.

The top and bottom of the elevator represent the top and bottom of your file. The further text is into the file, the further down the left border the elevator starts. The more of the elevator is in the left border, the more of the file is visible in the window (hence the shorter the file). As you scroll through the file, the highlighted portion of the left border moves up or down the left border.

5.2.4 The Corners

Each window corner has two commands that are invoked with either [■ □ □] or [□ □ ■] on that corner.

[■ □ □] on a corner **tops** the window, unless it is already the top window, in which case it *bottoms* the window.

[□ □ ■] on a corner lets you stretch that corner by *dragging* it. An elastic shadow border follows the mouse cursor and the window assumes the size of the shadow border when you release the [□ □ ■] .

The corners are *functionally* separate from the banner line or any of the border lines.

5.3 The Text in your POINT Window

5.3.1 Displayed Text

Text in the window is a representation of the contents of the file in your window and is displayed as it would be on a printer. In most text files, lines end with a **CR-LF (Carriage-Return, Line-Feed)** sequence (**ASCII 13 and 10**). This sequence serves to position text that follows the **CR-LF** at column 1 on the next line in the window.

5.3.2 Displayed Characters

Characters are displayed using the graphics symbols defined by the *IBM PC* for the 256 possible character values, with these exceptions:

- ⇥ moves to the next tab stop and is not displayed as a tab character.
- **Line-feed** (**ASCII 10**) shows as a blank, and positions following text onto the beginning of the next line.
- **Carriage Return** (**ASCII 13**) immediately followed by a **Line Feed** is ignored. A **Carriage Return** not followed by a **Line Feed** is displayed normally.

5.3.3 The End-of-line Character

At the end of each line, *POINT* displays a vertical rectangular blank that represents the **Carriage-Return/Line-Feed** sequence. This blank is only visible if you include it in a selection. **Line-Feed** (**ASCII 10**) alone will ba accepted as an end-of-line character, but when you press ↵ , a **Carriage Return/Line Feed** is inserted there in the file, and is called the "end-of-line character."

5.3.4 Selected Text

There is always a *current selection* (unless no windows are open). The selection is a sequence of one or more characters in one of the displayed files. It's not necessary for the selection to be visible in a window (it might be in a window partly covered by another window).

Selection specifies the text that you want to manipulate. After selecting, you generally issue a command that affects the selection.

Selection has no effect except to change the appearance of the display until you issue a command that affects the selection. So, making a selection does not imply any commitment to do anything. And, if you select the wrong item, simply make another selection before issuing a command.

The `OPTIONS` menu has a `textColors` option which sets various color or video attributes of both the text in a window and of selected text. A contrasting color can help keep track of selected text for foolproof manipulation.

5.3.4.1 Select Text

To make characters in the document the current selection, do this:

Press (■ □ □) to start the selection. Hold (■ □ □) down and move the mouse to select additional text. You can move either forward or backward from where you first pressed the (■ □ □).

The selection starts in **character mode**. If you release the (■ □ □) and press it again inside the selection, you select **word mode** and the selection is extended one word at a time. A third press selects **line mode**, and the selection is extended one line at a time. A fourth click returns to **character mode**. Selection mode affects how *copy* and *move* act.

[*#26*, ((■ □ □) , *inside window*)]

5.3.4.2 Select Words and Lines

Click [■ □ □] once for *character mode.* If you move the mouse while [■ □ □] is still pressed, the cursor extends the selection *character by character.*

If the mouse cursor is on a currently selected character and you press [■ □ □] *twice*, you select *the entire word.* In this case the cursor extends the selection *word by word.*

A third click inside the selection changes the selection to *line mode*, and the cursor extends the selection *line by line.*

A fourth click cycles back to *character mode.*

5.3.4.3 Extend the Selection

[□ □ ■] extends or contracts the selection. To start, click [■ □ □] to select. Then, move the mouse cursor to where you want the highlighted selection to end. Press [□ □ ■] and the selection will extend to that point. If you hold down [□ □ ■], the highlighted selection will follow the mouse cursor just as it does when you select with [■ □ □].

You can move either forward or backward from the place you first pressed [□ □ ■]. To extend a selection beyond the immediate text in the window, start with [■ □ □], move the window as needed, and extend the selection with [□ □ ■].

Characters are added to (or removed from) the current selection. Press [□ □ ■] to start the extension. Hold the **button** down and move the mouse to select more or less text.

[*#39*, **mouse** ([□ □ ■] , *inside window*)]

5.3.4.4 Select the End-of-line

The *end-of-line* character is at the end of each line in the window. It appears as a blank on the screen, but can be distinguished from actual spaces in the text since it is always *the last character on a selected line.* Actually, **Carriage-Return/New-Line** (the combination) marks the end of a line in *DOS* formatted text. You can't select either **Carriage-Return** or **New-Line** separately if they are together.

A **New-Line** alone is an end-of-line character. In this case you can effectively select the **New-Line** alone. **Carriage-Returns** alone do display and can be selected.

Deleting the *end-of-line* character joins that line with the next line.

5.3.4.5 Delete Text

Selected text is deleted from the screen and saved in the scrap buffer for later insertion. The selection mode of the text in the scrap buffer is recorded.

[*#4*, Del, F1]

5.3.4.6 Delete Text, Not to Scrap

Selected text is deleted. The scrap buffer is not changed.

[*#61*]

5.4 Input from your Keyboard to POINT

5.4.1 Insert Text Characters

5.4.1.1 The Insertion Point

The insertion point is *just before* the first character of the selection. All characters typed on the keyboard are *inserted* here. First, select an insertion point in the text with the mouse cursor. Then click the ■ □ □ . In this case, you are **not** selecting text to alter or manipulate; but rather a **point** at which you will enter new text.

Most of the 128 **ASCII** characters are inserted by typing a corresponding key.

Control characters are not inserted, but reserved by the operating system or interpreted specially by the editor. These include:

Ctrl-2 (nul)
Ctrl-C (like Ctrl-Break)
Ctrl-H or ← Back
Ctrl-M or ↵ (carriage return)
Ctrl-[or Esc (escape)
Ctrl-P , Ctrl-S (suspend)

If ↵ is pressed and `autoIndent` is set to `1`, the next line is indented to the same level as the previous line. In fact, it will be indented with the same sequence of spaces and tabs that began the previous line.

If `overType` is set to `1`, the next keystroke replaces the character after the insertion point.

5.4.1.2 Insert and Overtype Modes

You are initially in insert mode: any printable character you type appears at the insertion point.

Switch to `overtype` (or overwrite, replace, or typeover) mode, from the `OPTIONS` menu or by pressing Alt-O . Now each new character *types over* existing text and *replaces* the next highlighted character.

The banner line shows `OverType` to indicate that you are in `overtype` mode.

5.4.1.3 Backspace over Text

← Back erases the *character* previous to the selection.
Ctrl-← Back erases the previous *word*.

A *word* to Ctrl-← Back is either:

- A sequence of *letters* (upper and lower case) and *numbers*
- A sequence of *special characters* (non-alphanumeric)

The white space (spaces, tabs, and the end-of-line character) after the word is also deleted.

Ctrl-← Back begun in the middle of a word deletes only the characters *before* the selection.

5.4.1.4 Insert ASCII Characters

You can insert any *extended ASCII* character (the 256 characters defined for the *IBM PC*) into the text. Specify the character with its numerical *ASCII* value in either *decimal*, *hexadecimal*, or *octal*.

If the leading two characters of the value you give are 0X the rest of the number is interpreted as *hexadecimal*.

If the first character is 0 (and the second is *not* X), then the number is interpreted as *octal*.

If neither is the case, the number is interpreted as decimal.

[*#23*, Alt-A , **EDITING** menu]

5.4.2 Enter a response

5.4.2.1 Respond to a Prompt

Sometimes a prompt appears on the bottom line of the display. Type text normally for a response. ← Back erases the last typed character, and Ctrl-← Back erases *everything on the line*. You can't select a response when the prompt is displayed, but □ □ ■ can copy *what is already selected* into the response area.

Conclude the response to a prompt with ↵ or ■ □ □.

Esc can be used to cancel an action that has requested input.

Sometimes a default response appears. To accept this response, press ↵ or ■ □ □. ← Back erases *the last character* of the default response and lets you edit the default response. *Any other key* erases the default response and starts a new response with the key you pressed.

Some responses only need a yes or no answer and terminate after you type either Y or N. For **Yes/No** questions, ■ □ □ is **Y** and *any other mouse button* is **N** *only* if the mouse cursor is on the bottom line of the screen. **Yes/No** questions also have default responses (usually **Y**).

5.4.2.2 Select a File Name

When you load a new file into a window or create a new window, you are shown a list of file names. Select a name by clicking on it with the mouse.

The top line shows information about the files displayed:

- The current directory.
- How file names are sorted.
- Name pattern for displayed file names.
- Number of files in the directory.

If there's not enough room to show all the file names, you see the message `SOME FILE NAMES MISSING` and only *some* of the file names from the file pattern.

The second line of the display shows five commands that affect file name display. `Cancel load` cancels the load or window create.

If the list of files doesn't fit on one screen, review other file names by using `Next page` and `Previous page`.

Click on **New pattern** to specify a pattern used to display the list of files. To specify more than one pattern in the list, separate each pattern with a vertical bar character (|). A file pattern can include a drive letter and a **PATH** name. The file name by itself, or as the last component of the path specification, can include ***DOS*** wild card characters (* and **?**).

Here are some examples of file patterns:

.	all files in the current directory
*.DOC	all files with the extension .DOC
.DOC |.TXT	all files with .DOC or .TXT extensions
\BIN\A*.BAT	.BAT files in \BIN beginning with A
\BIN*.BAT |\TEST*.BAT	.BAT files in \BIN and \TEST

From this screen you can:

- Click on the file name you want to load,
- Type in a file name directly, or
- Click on a command name to change the display.

5.4.2.3 Options

OPTIONS lets you change many options interactively. If you select an option (except for a **True/False** option), you are asked for a new value. **True/False** options are *toggled* from true to false, or false to true.

5.4.2.4 Set Screen Colors

If you choose any of the **color** options, you are shown a screen that lets you temporarily change the color scheme for the active window. The choices you make are immediately reflected in the sample screen in the box at the bottom of the screen. For permanent changes you must change the color statements in PT.INI.

5.5 File Handling in POINT

5.5.1 Files in Windows

All windows show a file name on the banner. When the window is loaded, it displays the contents of the file. As the file is edited, the window shows the most current version, but the original file on disk is unchanged until you save the edited file.

All new text in the window is kept temporarily in a work file called PTTEMP.XXX. This file is in the working drive and directory, but it can be reset with the `workDrive` option in PT.INI.

You can put the work file on a RAM disk, but it won't speed things up much unless you are editing a very large file or many different files. The reason is that *POINT* tries to keep all the active parts of the work file in its internal buffers anyway. With large files or with many files, it will not be able to do this.

↵ by itself, when prompted for a `New Window` file name, creates an empty scratch window with the name `UnNamed.x`.

5.5.2 Backup Files

When you save a file, the previous version is saved in a .BAK file which contains *the version you first loaded*, not the version before the last save. That is, even if you save the file two or more times during an editing session, when you quit *POINT* the .BAK file will contain the version of the file before you began your session.

The `makeBaks` option determines whether .BAK files are generated at all.

5.5.3 File Handles

POINT has a number of files open at the same time; therefore it needs sufficient ***DOS*** file handles to be able to have many files in many windows. You can tell ***DOS*** how many file handles to allocate in the CONFIG.SYS file. We recommend that you specify at least **20** and preferable **30** file handles in CONFIG.SYS.

5.5.4 Feedback On Long Operations

Replace and **Write** operations now provide feedback about their progress as they execute. The idea is that any operation that might take more than a few seconds will provide feedback on its progress.

Replace tells you what percent of the operation has been completed so far. This is true of the **replace with verify** and the **global replace**. The percent feedback tells you how much of the file has been searched so far, not what percent of the replaces have been done. (It cannot know how many actual replaces are needed until the replace is completed.)

Any file that writing command provides continuous feedback about what percentage of the operation has been completed. This applies to **write**, to **same**, and to all file writing resulting from **quit** commands.

5.5.5 Read-Only Files

There is now a facility for *read-only* files. One purpose of this is to correctly handle files that have *read only* permission in ***DOS***. Another purpose is to allow you to edit files without the possibility of accidentally changing them. A file in a window can be made *read-only* in three ways.

- Load a file marked by ***DOS*** as *read-only* into a window.
- Load a file into a window when the global "readOnly" option flag (a new option) is true.
- Use a new command that toggles the readOnly state of a file in a window.

`Toggle Read Only` [command #86] changes the **`readOnly`** status of the file in the active window. It does not let you change the **`readOnly`** status of a file marked *read-only* by ***DOS***.

You can edit a *read-only* file on screen only. The changed version cannot be saved. Trying to save a **`readOnly`** file will result in an error message. Trying to quit with a **`readOnly`** file that has been edited will generate an error message and give you a chance to cancel the quit (by pressing the Esc key) or to proceed with the quit without saving the changes (by pressing any other key).

You can write a **`readOnly`** file to another filename.

Chapter 6
POINT Commands

6.1 Invoking POINT

Here is the format for invoking the *POINT Editor* from the *DOS* command line:

PT [/O] [/H] [/V] [FILE1 [FILE2 ...]]

/
– / *slash* or – *hyphen* precedes option letters.
–H, –O, and –V, or / H, / O, and / V designate options.

/O Overlaid is the initial windows default, with each window *full screen* and *overlaid.*

/H Horizontal split loads initial windows full-width, with the first screen full, the second a few lines down, etc.

/V Vertical split is similar, except that initial windows are *all* full screen height and spaced vertically and evenly across the screen.

/L *POINT* looks for a file in the current directory named PT.LAS, from which it can reconstruct the last editing session. (Of course, you will not be able to undo edits from the previous session.)

[FILE1 [FILE2 ...]]

POINT creates a window for each file name on the command line. If the file name you specify contains wildcard characters (* or ?) it will expand to contain all matching file names, as specified in the *DOS* manual. A maximum of 20 files is allowed.

Several additional options relate to invoking *POINT*.

`initialWindows` sets arrangement of the initial windows

`buffers` sets buffer space and memory to be used.

`videoMode` sets how the screen will be updated.

`workDrive` tells where the work file will be placed. A RAM disk is useful here for large files.

6.2 Command Format

The commands in this chapter are grouped by function type.

At the end of each command description [square brackets] contain the ways to invoke the function. The following parameters are used within the brackets.

#n — A number (e.g., *#7*) gives the number which identifies the command to the top line, to a keystroke, to a mouse button, or to a menu.

(□ □ □ , *position*) — □ □ □ , followed by *where the function is called,* is in parenthesis.
(■ □ □ , *any corner*), for example, means:
press ■ □ □ on *any corner of a window.*

? or Ctrl-? — Keys and key combinations that invoke commands.

<command> — Top line commands (e.g., **`find`**) show the command name from the top line.

<MENU> — Menu commands (e.g., **`WINDOWS`**) show the menu name from the top line.

Top line command and menu names are those used in the distributed files PT.INI and COLOR.INI. PTEXPERT.INI uses a different top line.

6.2.1 Esc and Ctrl-Break

Esc is a general escape. Most commands can be aborted with no action or change by hitting Esc when keyboard or mouse input is requested. Esc only stops actions that use keyboard input (e.g., writing a file, searching for a text string).

Ctrl-Break stops the current editor action.

6.3 Window Management Commands

6.3.1 Create a New Window

To create a new window, click ■ □ □ on **New Window** from the **WINDOWS** menu. You are asked to specify any two opposite corners. Press (and hold) ■ □ □ at one corner. The first corner can be any of the four corners of the new window position. An elastic window border will follow the mouse cursor. Release the button at the opposite corner. If you double-click at the first corner without moving the mouse, the opposite corner will be taken to be the lower right corner of the screen.

Next, ***POINT*** asks for a file to be loaded into the window. Select the file from a menu of file names or type in the file name. Esc aborts the action, and no window is created. ↵ creates a window for a file named UNNAMED.X, where X is an alphabetical character [A .. Z].
[*#7,* Alt-N]

6.3.2 New Window From Selection

This is exactly like the **New Window** command (above), except that the file to load into the new window is taken from the text you have selected.
[*#82,* Alt-F10, Ctrl-F10]

6.3.3 Hide Window

The window is no longer displayed but is still open. The window is then "hidden" and only shows up on a second list at the end of the **TOPLIST** menu. Selecting a hidden window from **TOPLIST** unhides it and makes it the top/active window. It will be the same size and in the same position as when it was last visible.
[*#38,* Alt-X]

6.3.4 Close Window

The window is removed from the screen. If this is the last window that shows the file, you are asked whether to save the edited version of the file or to discard the edits and leave the original file unchanged. The file is closed and the window is deleted.
[*#17*, Alt-C]

6.3.5 Close Window and Save File

The window is removed from the display. If the file has been changed it is saved automatically (no verify is requested). The file is closed and the window is deleted.
[*#72*]

6.3.6 Split Window

A window can be split (vertically or horizontally) into two independently scrolled windows on the same file. The original window is unchanged and the split window is on top of it. The split window starts where the split was made and goes to the right or lower border of the original window. The `tiledSplit` option causes the original window and the new window to each take up part the space occupied by the original window.
[(❑■❑ , *top or right border*)]

6.3.7 Change Color Combination in Window

This command changes the text and border colors of the active window to the next color combination on the list in the `textColors` and `borderColors` option. After the last one is used, the command cycles back to the first one. The default color combinations is included in this cycle.
[*# 87*]

6.3.8 Window Positioning

6.3.8.1 Top Window

The window is made the top window shown on the display.
[*#12*]

6.3.8.2 Bottom Window

The window is made the bottom window.
[*#29*, Alt-B]

6.3.8.3 Top/Bottom Window

The window is made the top window unless it is already the top window, in which case it is made the bottom window.
[*#28*, (■□□ , *any corner*)]

6.3.8.4 Zoom Window

The window is expanded to cover the entire screen, or if it is already zoomed, it is returned to its original position.
[*#11*, Alt-Z , **WINDOWS** menu]

6.3.8.5 Change Window Size

Respecify the size and position of the window. Press (and hold) the ■□□ at any corner. From there, an elastic window border (where the mouse cursor is the opposite corner) will follow the cursor. Release the **button** when the window is the desired size. If you double-click at the first corner, the opposite corner is taken to be the lower right corner of the display.
[*#13*, **WINDOWS** menu]

6.3.8.6 Stretch Window

Change the window size on the display. The corner you first pressed the ■ □ □ **mouse button** on is moved to the mouse position where you release the **button**. An elastic window border will follow the mouse as it is moved. The size can be increased or decreased. The position of the opposite corner of the window is not changed.
[(□ □ ■ , *any corner*)]

6.3.8.7 Move Window

The window is moved on the display. With the mouse cursor on the banner line, press and hold the □ □ ■ . A shadow border follows the mouse. Release the **button** when the window is at the desired position. The size of the window is not changed unless you release the button while the shadow border is smaller due to being moved against the side of the display.
[(□ □ ■ , *banner line*)]

6.3.8.8 Exchanging The Top Two Windows

This command exchanges the places of the top two windows. This is useful when you are changing between two windows frequently but you also have other windows open also.
[*#88*]

6.4 File Management Commands

6.4.1 Load File

Loads a file into a window. A menu of file names is displayed. Use ■□□ to select a file name, or type in the name. Esc cancels the load. ↵ by itself creates a **New Window** with a file named Unnamed.x. If the file you ask for does not exist, you are asked whether or not to create it. If you answer N , the file load is canceled. If **autoCreate** is 1, then the file is created automatically without verification. If the file currently in the window was changed and is not displayed in any other window, you are asked if you want to save the edited file.
[*#33*, Alt-L , **WINDOWS** menu]

6.4.2 Load File From Selection

This is like the load command, except that the file name is taken from the selection.
[*#73*, ⇑Shift-F10]

6.4.3 Save File

Writes the edited version of the file, and releases the editing space. This action *cannot be undone*, but a .BAK file is generated with the old version of the file. If **makeBaks** is set to 0, no .BAK file will be created. You cannot undo changes if you have saved the file since the changes were made.
[*#54*, Alt-S , **WINDOWS** menu]

6.4.4 Save All Unsaved Files

All files that have been *edited but not saved* are saved as described in the save file command (above).
[*#46*]

6.4.5 Write With New Name

Writes edited version of a file to a new file. You are prompted for a new file name. This action cannot be undone.
[*#18*, Alt-W , **QUIT&ETC** menu]

6.4.6 Toggle Read-Only Status of File

This command changes the read-only status of the file in the active window, that is, it changes it from read-only to read-write or from read-write to read-only. It does not let you change a file to read-write if the *DOS* file permissions are *read-only.*
[*#86*]

6.4.7 Scroll Down

Moves window toward the end of file. If the window contains *N* lines, it scrolls *N-2* lines . If you do this with the mouse, the amount scrolled depends on how where the mouse cursor is on the left window border. More precisely, the text line next to the mouse cursor becomes the top line of the window. If the mouse cursor is on *line 1* in the window, the window scrolls one line. The **rightBack** option reverses the scroll direction of the mouse button.
[*#15*, PgDn , (■ □ □ , *left border*)]

6.4.8 Scroll Up

Moves window toward the top of the file. The window is scrolled *N-2* lines if the window contains *N* lines. If you invoke this with the mouse, the amount scrolled depends on how far down the left window border the mouse cursor is when you invoke the command. The window scrolls up so that the top line moves down to the mouse cursor. The **rightBack** option reverses the scrolling direction of the mouse button. If the **rightBack** option is 2 or 3, the text line next to the mouse cursor becomes the bottom line of the display.
[*#14*, PgUp , (□ □ ■ , *left border*)]

6.4.9 Scroll Left

Move the window to the left.
[(■ □ □ , *bottom border*)]

6.4.10 Scroll Right

Move the window to the right.
[(□ □ ■ , *bottom border*)]

6.4.11 Thumb Vertical

Move window to a specific area in the document. Imagine the document laid out along the left border. The mouse cursor indicates where in the document you want to go. The window will be positioned so that the top of the elevator (the left border highlight) starts where the mouse cursor is when you thumb. Corners are not part of the left border and cannot be used for thumbing.
[([□ ■ □] , *left border*)]

6.4.12 Thumb Horizontal

Move window left or right depending on where on the bottom border you click.

[([□ ■ □] , *bottom border*)]

6.4.13 Search For String

A prompt is displayed for the search string. If the selection is in the window, a search is made from the point of selection to the end of the file. If the selection is in another window, the entire file is searched. The search ignores case unless `ignoreCase` is set to `0`. The default search string is the last string searched for. This string is automatically inserted as a response. Pressing [↵] accepts this as the search string. If you press [← Back] first, it erases the last character of the default string and lets you continue editing with more backspaces or new characters. Pressing any other key first erases the default string and starts a new string.

A find is displayed in character mode. If the string is already in the window, the window is not changed. If the string found is not in the window, the window is changed to show the string found on the third line of the display. The window is topped when the string is found only if the `topOnFind` option is set to 1. The `searchMode` option determines whether the search proceeds forwards, backwards, or circularly.

In a search string the following escape sequences are valid:

" `\n` " represents the end-of-line character (*ASCII* 13 followed by *ASCII* 10)
" `\N` " represents the line feed character (*ASCII* 10)
" `\r` " represents the carriage return character (*ASCII* 13)
" `\\` " represents a single backslash character (\ )

[*#19*, [⇑Shift]-[F6] , **MOVING** menu, **find** on the top line]

6.4.14 Search For Selection

This is the same as search for string, except the current selection is used as the search string. The selected search string can include one or more end-of-line characters. You can select a string in one window and search for it in another window.
[*#27,* F6 , `MOVING` menu, `next` on top line]

6.4.15 Search Backwards

This is like search for selection, except the search goes in the opposite direction. Unless `searchMode` is set to `1`, this is from the selection to the beginning of the file.
[*#66,* Alt-F6 , Ctrl-F6 , `MOVING` menu, `prev` command on top line]

6.4.16 Replace String

Replaces occurrences of a string with a different string. The string to search for is requested first. Then, the string to replace it with is requested. Both search and replace strings can contain the escape sequences `\n` , `\N` , `\r` , or `\\` as described in the search command (above). Next, you are asked whether the replace operation should take place only within the current selection or from the current selection to the end of the file (a "global replace"). Finally, you are asked to indicate whether you want to verify each replacement or replace them all without verification. If you choose to verify each change, Y makes the replacement, N does not replace, but goes on to the next occurrence of the string, and Esc terminates the replace operation.
[*#20,* Alt-R , `EDITING` menu]

6.4.17 Beginning of File

Positions the window at the beginning of the file.
[*#8,* Ctrl-PgUp]

6.4.18 End of File

Positions the window at the end of the file, with the end-of-file marker on the bottom line of the window.
[*#9,* Ctrl-PgDn]

6.4.19 Beginning of File and Move Selection

This command works like the Beginning of File command but it also moves the selection to the first character of the file. It is can be used before a search or replace that you want to start at the beginning of the file.
[*#78*]

6.4.20 Go To Line Number

You are prompted for a line number. The window is repositioned so that the requested line number is the top line in the window.
[*#16*, Alt-G , **MOVING** menu]

6.4.21 Go To Selected Line Number

The selection is taken as a line number. Nondigits at the beginning of the selection are ignored. The window is repositioned so that that line number is the top line in the window.
[*#84*, Alt-F9 , Ctrl-F9]

6.4.22 Go To Last Place

The window is repositioned to the last place you came from with a nonrelative motion (that is, not a scroll). This is the last place you jumped from with one of these commands: go to line number, go back to last place, search, beginning of file, or end of file. A separate "last place" is remembered for each window.
[*#56*, F3 , **MOVING** menu]

6.4.23 Go To Selection

The window containing the selection is made the top window and repositioned so that the selection is near the top of the window.
[*#58*, ⇑Shift-F3 , **MOVING** menu]

6.4.24 Find Matching Bracket

The bracketing character that matches the first selected character is searched for. The allowed bracketing characters are (,), [,], {, and }. The file is searched in the correct direction and the matching bracket is found, taking nesting into account.
[*#83*, ⇑Shift-F8]

6.5 Menu Commands

To use pull-down menus. move the mouse cursor to the top line. Then press and hold ■ □ □ on the (upper case) menu name. When the menu appears, select a *menu option* with the mouse cursor. In the menu, the command is the color of the selected text.

If you release the ■ □ □ on a command string, that command is carried out. If you release the mouse button outside the menu, no command is executed.

Esc exits from a menu without selecting anything. ↵ executes a command in the same way as a mouse button click.

6.5.1 Top line Commands

Commonly used commands are listed on the top line. Click ■ □ □ on one of these to be execute the command . It is possible to put any command on the top line (and give it any name) by modifying PT.INI.

PT.INI also lets you have:

- *Several top lines*, each accessed by a different button
- *A bottom line* of menus and commands instead of a top line
- Pop-up mouse menus.

6.5.2 User Menus

These commands display the eight user-defined menus and allow you to select a command from them. They can be displayed as top line menus, bottom line menus, drop-down menus (from the top line), or as pop-up menus.

[*#50* (1), *#51* (2), *#52* (3), *#60* (4), *#65* (5), *#67* (6), *#79* (7), *#80* (8)]

6.5.3 TOPLIST Menu

A menu is displayed that lists the file names of the files in all open windows. The file names are shown from the top window to the bottom window. If there are hidden windows, they are listed after the list of visible windows (a double line divides the two lists). Selecting file name from the `TOPLIST` menu will top that window. Selecting the file name of a hidden window unhides and tops the window.
[*#53*, Alt-X]

6.5.4 OPTIONS Menu

A menu of ***POINT*** options is displayed. The option to be changed has to be selected Options are changed in two ways. **`True/false`** options toggle from true to false to true when selected. *Numeric* or *string* options display a prompt for a new value. The default input is always the current value of the option. Esc exits option setting with no changes.
[*#63*]

6.5.4.1 Interactive Color Settings

`textColors`, **`borderColors`**, and **`msgColors`** option show the color settings. Each setting is two hexadecimal digits.

`textColor` Shows the color setting for the text and the selection.

`borderColor` Shows the color setting for the banner line, the border and the elevator.

`msgColor` Shows the color setting for the Info Message, User Input,
Error Message, and Topline.

Selecting **`textColors`**, **`borderColor`** or **`msgColor`** puts you in color setting mode where you can set the colors of the active window or any of the other ***POINT*** color settings. Esc exits color setting mode with no changes.

The color setting mode will display a screen with three major parts.

The top is a menu line with 10 commands. Nine of these are used to change the color of the display. The part that will be changed is selected by clicking on its name on this menu line. The tenth command exits interactive color setting.

The second part of the screen is a line of four commands and a menu of the 128 possible combinations of foreground and background colors you can select from. Select a color combination by clicking on it. The commands allow you to look at another 128 color combinations (the alternate colors) which will either be blinking or have bright background colors. Two of the commands switch you back and forth between the plain and alternate color sets. The other two commands change the alternate color set from blinking to intense and back again.

NOTE

Commands to switch between blinking and intense colors reprogram the hardware: their effect will persist until you turn off your machine.

The third part of the screen is a sample window showing how the selected color combination will look, and samples of the four other display parts.

Pressing Esc while in color setting mode returns you to editing without making any color changes.

Click on the `Exit` menu item on the top part to exit the color selection mode.

6.5.4.2 Redefine Keys, Buttons and Mouse Motions

This lets you interactively change the definition of a key, mouse button or mouse motion subcommand. It is useful when you need an easy-to-issue command. For example, you might want to change F3 to issue the **change case** command so that you can quickly change the case of a sequence of words.

The feature is accessed through the **`OPTIONS`** menu item **`Redefine...`** When you select **`Redefine...`** you are asked what type of action you want to redefine. Redefinition options are:

`k` a key

POINT asks you to press the key you wish to redefine. Then it asks you for the new command to assign to that key. This process is described below.

`b` mouse button

POINT asks you to press the mouse button you wish to redefine. Then is asks you for the command to assign to that button. You can also redefine a combination mouse button ⇑Shift , Alt , or Ctrl . (Do this in PT.INI also.) Just press ⇑Shift , Alt , or Ctrl and then the click the mouse button.

`1` subcommand of the first mouse motion command

`2` subcommand of the second mouse motion command

`1` or `2` prompts you for the subcommand direction. The options are a compass point (n, ne, e, se, s, sw, w, or nw) followed by the ↵ key, or the ↵ key alone to redefine the *no motion* command.

After you select the action to be redefined, ***POINT*** displays directions on line 24 of the screen, and a command number and command description on line 25. These are the same one-line descriptions you see if you have the `helpMode` set to `1` or `2`. The first command you see is the command currently assigned to the action you specified.

You can scroll up and down through all the possible commands (presently there are about ninety (90) ***POINT*** commands) with the [↑] and [↓] cursor keys. You can jump directly to a command by typing its number. A one digit command number must be preceded by a `0`, or followed by the [↵] key.

When the command you want is showing, press [↵] to complete the redefinition.

6.6 Mouse Motion Commands

Mouse motion commands make it possible for you to execute a specified command by simply moving the mouse in one of eight directions on your desk/screen.

6.6.1 Issuing a Mouse Motion Command

The mouse motion command is assigned to a mouse button. The mouse motion command has nine associated subcommands. You issue a subcommand by pressing the mouse button, moving the mouse at least one character in one of the eight compass directions, or not move at all and then releasing the mouse button. The directions are:

`N`	north or up
`NE`	north-east or up and right
`E`	east or right
`SE`	south-east or down and right
`S`	south or down
`SW`	south-west or down and left
`W`	west or left
`NW`	north-west or up and left
—	no movement

POINT only looks at where you press the mouse button and where you release it to determine which direction you went.

While the mouse button is depressed, *POINT* gives you feedback as to which command it will execute if you were to release the button at that point. The feedback is an arrow pointing the direction you have moved (or a single dot for the no motion command). If you have `helpMode` set to `1` or `2` then a one line description of the command is given on the bottom line of the display. It is recommended that you set `helpMode=1` or `helpMode=2` while you are learning to use the mouse motion commands.

A ninth subcommand is the no motion command. It is issued if you release the mouse button at the same point as you pressed the button. This command is easily issued as a fast click of the mouse button. Therefore this should be the most commonly used command of the nine mouse motion subcommands.

6.6.2 Canceling A Mouse Motion Command

Pressing a second button cancels the mouse motion command.

6.6.3 Default Mouse Motion Subcommands

You can specify the nine subcommands of the mouse motion command (see section 3.3.5 for information on how to do this). The default commands are:

```
              previous
        undo      |     delete
            \     |    /
             \    |   /
copy <----- duplicate -----> move
             /    |   \
            /     |    \
        redo      |     insert
                next
```

The command numbers are:

duplicate	*#2*
previous	*#66*
delete	*#4*
move	*#70*
insert	*#55*
next	*#19*
redo	*#24*
copy	*#69*
undo	*#25*

These rules will help you remember these commands:

- downward strokes are positive (redo, next, insert)
- upward strokes are negative (undo, prev, delete)
- they are paired, down and up (redo/undo, next/prev, insert/delete)
- diagonal strokes make editing changes
- horizontal strokes do copy and move
- vertical strokes do searching

6.6.4 Second Mouse Motion Command

The second mouse motion command is for people with a three button mouse who want to assign two mouse buttons to mouse motion commands.

[*#46*]

The default subcommands of the second mouse motion command are:

```
              move to
             last place
     close        |       delete-
     window       |      no scrap
            \     |     /
             \    |    /
  copy to <--- extend ---> exchange
   scrap       /  |  \       with scrap
              /   |   \
             /    |    \
      bottom      |     cancel
      window      |
               TOPLIST
```

The command numbers are:

extend	*#39*
move to last place	*#56*
delete/no scrap	*#61*
exchange with scrap	*#62*
cancel	*#22*
TOPLIST	*#53*
bottom	*#29*
copy to scrap	*#57*
close window	*#17*

The second mouse motion commands can be changed in PT.INI with lines of the form:

`nN=commandNumber`

where **`N`** is 0, 1, 2, 3, 4, 5, 6, 7, or 8 as in the chart above.

6.7 Editing Commands

6.7.1 Copy

There are several ways to *copy* text: the `copy` command, the `duplicate` command, and via the *scrap buffer*.

6.7.1.1 Copy

Select the text to be copied. Move the mouse cursor to the point where you want to copy the selection. Now, press ⇑Shift and click ■ □ □ . The text will be copied to the place where you release the ■ □ □ . If the *selection* is in *word* or *line* mode, it will be copied in front of the *word* or *line* where you release the mouse button.
[*#69,* ⇑Shift-■ □ □ **or** ⇑Shift-□ □ ■]

Step 1: Select the text you want to copy.

Step 2: Move the mouse cursor where you want to copy the selected text.

Step 3: Press ⇑Shift and click the ■ □ □ **mouse button.**

If you select the insertion point in character mode, then the text is copied in front of the character you are pointing at.

If you select the insertion point in *word mode* then the text is copied in front of the *word* you are pointing at.

If you select the insertion point in *line mode* then the text is copied *in front of the line* you are pointing at.

6.7.1.2 Copy To Scrap

This copies selected text into the scrap buffer. The mode (character, word, or line) of the selection is remembered and used when the text is inserted.
[*#57*, ⇑Shift-F4 , `EDITING` menu]

6.7.1.3 Copy with Scrap Buffer

This method uses the **scrap buffer** for insertion of text. Move the cursor to where you want the text copied, select that point with the ■ □ □ , and press Ins .

If the text in the scrap buffer was selected in *word mode* or *line mode*, then it will be inserted in front of the character, word, or line that contains the insertion point. You can insert the same text from the scrap as many times as you want.
[*#55*, Ins , F2]

6.7.1.4 Duplicate

Press ■ □ □ where you want to copy the duplicated text and press F4 . Then select the text to be duplicated and press F4 again.
[*#2*, F4 and ■ □ □]

Step 1: Select the insertion point for the text you want to copy (this is the reverse order from the `copy` instruction).

Step 2: Press F4 to go into `duplicate` mode.

Step 3: Now, select the text to be copied.

Step 4: Press F4 again. The selected text will be copied to the insertion point.

This ends duplicate mode. The new insertion point is the first character *after the copied text.*

While in `duplicate` mode you can execute commands, such as scrolling windows, topping windows, searching for text, creating new windows, etc.

Use `duplicate` to copy several things to one place.
Use `copy` if you already know where you want to insert the text.

An easy way to `duplicate` is to select a section of text and then press F4 *twice.* This is especially useful for duplicating one or more lines of text.

6.7.2 Move

You can move text with the `Move Text` or with the `Extract` command

6.7.2.1 Move Text

Select the text to be moved. Then move the mouse cursor where you want to move the text. Then while holding down Ctrl key, click ■ □ □ . Text is moved to the place where you release the ■ □ □ **button.** If selection is in *word or line mode*, the text is put in front of the *word or line* where you release the button.
[*#70*, Ctrl-■ □ □]

6.7.2.2 Extract

Press ■ □ □ where you want to place the extracted text, and press F5. Then select the text to be extracted and press F5 again.
[*#3*, F5 , and **mouse button**]

6.7.3 Redo and Undo

The last 50 editing actions in your current session are recorded in a *change history*, which is used by the `Undo` and `Redo` commands. `undoSize` determines how many changes are remembered in the change history (`50` is the default; `100` is the maximum).

Some actions cannot be undone or redone, such as the following:

> load file, write file, close window, search, redraw, change selection, change window size, scroll a window, move a window, top a window, bottom a window, or exit the editor.

Del by itself cannot be redone.
Thus, deletes are ignored when looking for the last action to redo.
[*#24*, F8]

6.7.3.1 Redo

Most editing actions can be redone. A redone action is recorded in the change history and can be undone.

The sequence "select text, delete, and type new text at the same point" is recorded as one action in the change history.

`redo` repeats the last action in the present environment.

`redo copy` copies the last text that was copied into the present insertion point.

`redo move` acts exactly like `redo copy`, since the moved text is already gone from its original spot, and presumably you do not want to move it again from where you just moved it.

6.7.3.2 Reversible Undo

Undo Last Edit reverses the last editing action. A *delete* is restored. A *copy* is deleted. A *move* is deleted and reinserted where it was originally.

If the last action was to select text, delete it, and then type new text all at the same point, then this action is undone as a unit, that is, *new text is deleted and old text is restored.*

The undoing change is recorded in the change history so a second **Undo Last Edit** undoes the first Undo. If the **undoBack** is set to 1, then **Undo Last Edit** acts like **Undo, Erase History**. You cannot undo changes if you have saved the file since the changes were made.
[*#25*, Alt-U , F9 , **EDITING** menu]

6.7.3.3 Undo, Erase History

This acts like **Undo** except that it doesn't record the change in the change history, and in fact removes the change being undone from the change history. Thus, a second **Undo, Erase History** undoes the change before that.

Undo, Erase History undoes up to 50 previous edits. The **undoSize** option determines how many previous edits are saved. The window will be moved to where the undo will take place before the change is made. You cannot undo changes if you have saved the file since the changes were made.
[*#40*, ⇑Shift-F9 , **EDITING** menu]

NOTE

If you use undo and then decide to undo *more than one* step you have to use **Undo, Erase History** *two extra times*: one to undo the undo, and a second to undo the first undo!

If you prefer **Undo, Erase History** as the default, set **undoBack** to **1**.

6.7.4 Keyboard Macros

6.7.4.1 Begin/End Recording Macro Keystrokes

This begins recording keystrokes in the keyboard macro buffer. Keystrokes are recorded until this command is executed again. There is only one keyboard macro buffer.
[*#76*, Alt - M]

6.7.4.2 Play Back a Keyboard Macro

Keystrokes in the keyboard macro buffer can be played back at any time as if they were typed on the keyboard.

To define and use a macro, do the following:

Step 1: Execute `Begin/End Recording Keystrokes`. This records each keystroke that follows in the keyboard macro buffer. The keystroke commands are executed as you type them, so you are defining the macro by example.

Step 2: Execute `Begin/End Recording Keystrokes` again, to stop recording keystrokes and define the macro.

Step 3: Run the macro with `Play Back Macro`. This plays back keystrokes in the keyboard macro buffer just as you recorded them.

[*#77*, Alt - P]

NOTE

There is only one keyboard macro buffer, so if you record another macro, the previous one is lost.

Here is a keyboard macros to indent a section of code:

Step 1: Position the cursor on the first line to be indented,

Step 2: Press Alt-M .

Step 3: Press →| .

Step 4: Press ← (cursor key).

Step 5: Press ↓ (cursor key).

Step 6: Press Alt-M .

Step 7: Press Alt-P once for each line to be indented. The macro inserts the →| , moves to the beginning of the line, and down one line.

6.8 Quit Commands

The following actions exit *POINT* in a various ways, according to what is done with the files which have been modified during the editing session.

6.8.1 Quit-Ask About Files

Exits the editor. If any files have been changed but not saved, you are asked about each one. Press Y to save the new version of the file, or N to leave the original version unchanged. It will not accept any other keyboard input. You can also move the mouse cursor to the bottom of the screen and respond ■ □ □ for *Yes* and □ □ ■ for *No*. The editor is exited.

If you press Esc , the quit action is abandoned.
[*#5*, Alt-Q , QUIT&ETC menu]

6.8.2 Quit-Save Files

All files that have been changed but not saved are automatically saved. Exits the editor.
[*#48*, F2, QUIT&ETC menu]

6.8.3 Quit-Discard Edits

If files have been edited a single verify is requested. Exits the editor without saving files.
[*#49*, QUIT&ETC menu]

6.9 Execute DOS Commands

6.9.1 DOS Command Window

Selected text is taken as a ***DOS*** command and passed to the ***DOS*** command processor for execution. The standard output of the command is saved in a temporary file. When the command completes, a new window is created and the standard output of the command is displayed there. You can redirect either the standard input or the standard output of the ***DOS*** command on the command line (the selection). If you redirect the standard output the window created by the ***POINT Editor*** will be empty when the command completes.
[*#30*, **QUIT&ETC** menu]

6.9.2 DOS Command Shell

The screen is cleared, and a copy of the ***DOS*** command processor is executed. You can execute any ***DOS*** command using it. When you exit with the "exit" command, you are returned to the editor and the screen is redrawn. The output of the commands executed is not saved. The directory that was current when you invoked the ***POINT Editor*** is restored after you exit from the ***DOS*** command interpreter, but the current directory you were last in is restored if you invoke the ***DOS*** command interpreter again.
[*#32*, Alt-D, **QUIT&ETC** menu]

6.10 Miscellaneous Commands

6.10.1 Help

If **`helpMode`** is `1` or `2`, the help screen is shown for the last command you selected from the top line or from a menu (even if you did not actually execute the command). Otherwise you are shown the main help menu.
[*#64*, Alt-H]

6.10.2 Toggle 43-line Mode

Switches between *25-line* mode and *43-line* mode. The *43-line* mode is only available with the ***Enhanced Graphics Adapter.***
[*#68*, Alt-T]

6.10.3 Fill Lines To RightMargin

This fills lines so that each contains as many words as possible between column 1 of the text and the right margin defined by "rightMargin". "Word" means a string of characters separated by space characters. Space characters are spaces, tabs and end-of-lines.

When this command is invoked it performs this fill operation on all the lines that have any selected characters (it is not necessary to select the whole first and last lines, just one or more characters in each).

This command always uses column 1 as the left margin. If you want the block of text to be indented you can do this is three steps:

Step 1: Reduce rightMargin by the indent desired.

Step 2: Justify the lines.

Step 3: Indent the lines.

Indent lines with the replace command by replacing \n with <tab>\n (where <tab> is the tab character). This indents by one tab. Other indents are possible with similar replaces. Alternatively, you can use the keyboard macro facility to indent the lines one at a time.
[*#85*]

Appendices

Appendix A
POINT Features

Specifications

Maximum number of windows	20
Maximum number of open files	20
Maximum number of buffers	140
Minimum window size	3 rows by 10 columns
Maximum line size	any length
Maximum input string	100 characters
Maximum search string	50 characters
Maximum replace string	50 characters
Maximum number of menu characters	2500 characters
Maximum number of undos	100
Maximum disk buffers	300 K
User defineable menus	8

Notes:

Appendix B
Command Numbers

The following table shows command functions that can be attached to keys as defined in the PT.INI file.

The format for this definition is

```
k<N>=<Command Number>
```

where

`k` or `K` indicates that a key is about to be defined,
`<N>` is one of the key codes defined in Section B.1, and
`<Command Number>` is a number defined in Section B.2.

B.1 Key Codes

Extended Code	Characters
003	NUL
015	⇑Shift-→|
016-025	Alt plus Q W E R T Y U I O P
030-038	Alt plus A S D F G H J K L
044-050	Alt plus Z X C V B N M
059-068	F1 through F10 (unshifted)
071	Home
072	↑
073	PgUp
075	←
077	→
079	End
080	↓
081	PgDn
082	Ins
083	Del
084 - 093	⇑Shift-F1 through ⇑Shift-F10
094 - 0103	Ctrl-F1 through Ctrl-F10
0104 - 0113	Alt-F1 through Alt-F10
0114	Ctrl-PrtSc
0115	Ctrl-←
0116	Ctrl-→
0117	Ctrl-End
0118	Ctrl-PgDn
0119	Ctrl-Home
0120 - 0131	Alt plus 1 2 3 4 5 6 7 8 9 0 - =
0132	Ctrl-PgUp

B.2 Function Codes

	Number	Command
Window Management	7	New Window
	82	New Window From Selection
	38	Hide Window
	17	Close Window
	72	Close Window And Save File
	—	Split Window
	10	Redraw Screen
	87	Change Color Combination in Window
Position Window	12	Top Window
	29	Bottom Window
	28	Top/Bottom Window
	11	Zoom Window
	13	Change Window Size
	—	Stretch Window From A Corner
	—	Move Window
	88	Exchange the two Top Windows
Position File in Window	15	Scroll Down
	14	Scroll Up
	—	Scroll Left
	—	Scroll Right
	—	Thumb Vertical
	—	Thumb Horizontal
	19	Search For String
	27	Search For Selected String
	66	Search Backwards For Selected String
	20	Replace String
	8	Go To Beginning Of File
	9	Go To End Of File
	16	Go To Line Number
	84	Go To Selected Line Number
	56	Go Back To Last Place
	58	Go To And Show Selection
	83	Finding Matching Bracket
	78	Beginning of File and Move Selection

	Number	Command
File Commands	33	Load A New File Into Active Window
	73	Load Selected File Name Into Active Window
	54	Save File In Window
	46	Save All Unsaved Files
	18	Save File In Window Using Another Name
	86	Toggle Read-Only Status of File.
Select Text	26	Select Text
	39	Extend Selection
Insert Characters	—	Insert Character
	23	Insert ASCII Character
Copy And Move Text	69	Copy Text
	2	Duplicate Text
	70	Move Text
	3	Extract Text
	4	Delete Selection to the Scrap Buffer
	61	Delete Selection but Preserve the Scrap Buffer
	57	Copy Text To Scrap Buffer
	55	Insert Text From Scrap Buffer
	62	Exchange Selection with Scrap Buffer
	22	Cancel
	59	Toggle Insert/Overtype Mode
Redo And Undo Edits	24	Redo Last Edit
	25	Undo Last Edit (Reversible)
	40	Undo Edit (Multiple Step Undo)
Quit Commands	5	Quit And Ask About Unsaved Files
	48	Quit And Save All Unsaved Files
	49	Quit And Discard All Edits

	Number	Command
Execute DOS Commands	30	Execute Selection And Put Output In A Window
	32	Escape To A ***DOS*** Command Interpreter
Macro Commands	76	Begin/End Recording Keystrokes
	77	Play Back Recorded Keystrokes
Miscellaneous Commands	6	Set Value of the "debug" Variable
	21	Display POINT Information
	64	Invoke Help System
	68	Toggle 43-line mode (EGA)
	85	Fill lines to Right Margin
	81	Change case of a letter
Invoke Menu Commands	50	Invoke User Menu 1
	51	Invoke User Menu 2
	52	Invoke User Menu 3
	60	Invoke User Menu 4
	65	Invoke User Menu 5
	67	Invoke User Menu 6
	79	Invoke User Menu 7
	80	Invoke User Menu 8
	53	Invoke TOPLIST Menu
	63	Invoke OPTIONS Menu
Keyboard Cursor Movement Commands	34	Move Cursor Up
	35	Move Cursor Down
	36	Move Cursor Left
	37	Move Cursor Right
	42	Fast Cursor Movement
	43	Move Cursor to the Edges of the Window
	44	Move Cursor to the Edges of the Screen
	41	Simulate Mouse Buttons
	74	Move Cursor One Word (Blank Delimited) Left
	75	Move Cursor One Word (Blank Delimited) Right

Notes:

Appendix C
POINT Extensions

A *POINT* extension is a file which you can load and execute to extend the functionality of eh *POINT* Editor. *POINT* loads the extension on demand, and unloads it when it terminates.

An extension can be designed to remain resident in memory until space is needed to load another extension, or until you issue a command to free the memory. In this way often-used extensions do not need to be loaded again and again.

A **primary** extension may work together with another, **secondary**, extension. The secondary extension may be terminated independently of the primary extension. All extensions secondary to a primary extension will be terminated when the primary extension is terminated.

Calling POINT Extensions

Fifty-six (56) commands (#200 through #255) are available for calling ***POINT*** extensions. You can set them up in the PT.INI file as follows.

Assign a character string to any of these 56 commands. The character string begins with the name of the extension followed, where needed, by an argument. These arguments are passed to the command when it is executed. If the string ends with a space, the user is prompted for additional arguments to the command.

An extension is treated as primary if its name is immediately followed by an * (asterisk). An example of an extension is M2ASSIST, as described in **Appendix D**.

Appendix D
The M2ASSIST Environment

What M2ASSIST Does

M2ASSIST integrates the *POINT Editor* with the *LOGITECH Modula-2* system. It lets you check the *Modula-2* syntax of your program, compile it, look at compilation errors, link, and run, without exiting *POINT*.

These and other M2ASSIST functions are described on the following pages.

Check Syntax

`Check Syntax` checks the *Modula-2* syntax of text in the active window.

It highlights the *line of text* where the first error has occurred, and uses the bottom line of the screen to display the error message associated with the error in that line.

Compile

`Compile` lets you compile the file in the active window.

`Compile` begins by asking for compiler options.

You are prompted the last-used option. Esc lets you abort the request.

It saves the file in the active window and clears the screen.

It then runs the overlay version of the *LOGITECH Modula-2 Compiler.*

If it detects an error in the file, the compiler creates a .LST listing file. The source file is tagged with error messages from the listing file for examination with the `Next Error` function.

Current error messages are discarded if the window is closed, or if the file is saved or recompiled.

When compilation is done, any key or mouse button returns you to the editing session.

Find Next Error

`Find Next Error` highlights the next statement in the active window tagged with a compilation error message after invoking `Compile` or `Load Listing`.

You can insert/delete lines of text in the file and then go to the next tagged error.

Corresponding error messages from the .LST file show in a temporary error window at the bottom of the screen.

The error window closes at the first user action.

Link

`Link` links the .OBJ file associated with the active window.

`Link` begins by asking for linker options.

You are prompted for the last-used option. Esc lets you abort the request.

`Link` clears the screen and calls the ***LOGITECH Modula-2 Linker***.

When linkage terminates, pressing any key or mouse button returns you to the *POINT* editing session.

Run

`Run` executes the .EXE file associated with the active window.

When execution terminates, press any key or mouse button to return to the *POINT* editing session.

Load Listing

`Load Listing` reads the .LST file associated with the active window.

Error messages are tagged in the source file and can be looked at sequentially with `Find Next Error`, as explained above.

Load Templates

`Load Templates` loads the file M2ASSIST.INI, which contains templates for *Modula-2* constructs.

A template is a sequence of characters which can be inserted in the active window at the current cursor position.

The _ (underscore character) in the template tells the cursor where to appear after the template is inserted.

A template is identified by @ , followed by a letter in the set [a..z].

Here is a template from the current release of M2ASSIST.INI.

```
@h
(*
Title: _<title>
LastEdit:     <current date and time>
Author:       <author>
System:       LOGITECH MODULA-2 Version 3.0
(*
```

Text following the `@h` is inserted at the cursor position in the active window. The cursor is then repositioned *just before* `<title>`.

Quit M2ASSIST

`Quit M2ASSIST` releases the memory used by `M2ASSIST`.

Help

`Help` contains general information related to `M2ASSIST` functions.

Press any key or mouse button to return to the editing session.

How M2ASSIST Runs — About POINT Extensions

`M2ASSIST` is an extension of *POINT* (See Appendix C).

The current release of `M2ASSIST` actually consists of two extension modules: M2ASSIST.PTO and CHECKER.PTO.

M2ASSIST.PTO is a primary extension and CHECKER.PTO is a secondary extension. The functions of M2ASSIST.PTO are:

Argument	Function
0	Quit
1	Compile
2	Goto Error
3	Link
4	Read Listing
5	Run
6	Read M2ASSIST
7	Help
@x	Template

CHECKER.PTO has no arguments.

To run **M2ASSIST**, you must bind a key to a specific **M2ASSIST** function in the PT.INI file.

Consider the following lines in PT.INI:

```
c201="m2assist.pto 1"
k063=201
```

These lines bind function key F5 (**k063**) to command **201**, which specifies the **Compile** function from M2ASSIST.PTO. Note that M2ASSIST is defined as primary.

Analogously, to bind CHECKER.PTO to F2 , place the following lines in PT.INI:

```
c202="checker.pto"
k060=202
```

To bind a key to a *template* in M2ASSIST.INI, give the template name as an argument to M2ASSIST.PTO.

For instance, these lines

```
c210="m2assist.pto @x"
k064=210
```

insert template **@x** at the cursor position when you press F6 .

The distributed file PTM2.INI is an example of an initialization file set up to recall the **M2ASSIST** function.

Appendix E
Non-Mouse Reference

If you work on more than one system, you will occasionally work on one without a mouse. For this reason, a non-mouse interface is included here. Although non-mouse use is only occasional, we have made it as close as possible to the mouse interface.

Cursor keys move the mouse cursor, and other keys simulate mouse buttons.

Mouse button keys simulate *a completed click* rather than a *press* or a *release* by itself: you can't hold down a simulated mouse button.

Some mouse commands are simulated by two keystrokes on the numeric keypad on the right side of the keyboard.

Mouse Simulation Command Table

Key	Moves Mouse Cursor...
←	1 space left
→	1 space right
Ctrl-←	1 word left
Ctrl-→	1 word right
↑	1 row up
↓	1 row down

First Key	Next Key	Cursor Moves...
Gray +		
	←	10 columns left
	→	10 columns right
	↑	6 rows up
	↓	6 rows down
	PgUp	6 rows up and 10 columns right
	PgDn	6 rows down and 10 columns right
	Home	6 rows up and 10 columns left
	End	6 rows down and 10 columns left

First Key	Next Key	Next key moves *within window* to...
Gray −		
	←	Left edge of window
	→	Right edge
	↑	Top edge
	↓	Bottom edge
	PgUp	Top border
	PgDn	Bottom border
	Home	Left border
	End	Right border

First Key	**Next Key**	**Next key moves *within screen* to...**
Home		
	←	Left edge of screen
	→	Right edge
	↑	Top edge
	↓	Bottom edge
		Next key moves *within window* to...
	PgUp	Middle of window
	PgDn	Middle of window
		Next key moves *on the line* to...
	Home	Beginning of line
	End	End of line

First Key	**Next Key**	**Next key simulates mouse click...**
End		
	End	■□□
	↓	□■□
	PgDn	□□■
	←	⇑Shift-■□□
	Gray +	⇑Shift-□■□
	→	⇑Shift-□□■
	Home	Ctrl-■□□
	↑	Ctrl-□■□
	PgUp	Ctrl-□□■
	Gray −	Alt-■□□

Using Mouse Keys with a Mouse

Mouse simulation keys work differently if you use a mouse.

Cursor movement keys move the mouse cursor and the selection. This is normally what you want when you have a mouse since it saves additional keystrokes to move the selection after the mouse cursor is moved. Since the selection moves, the screen must be redrawn after each cursor movement command. This makes these commands somewhat slower.

If **cursorMouse** is set to **1**, mouse cursor simulation will only move the mouse cursor even if a mouse is present.

Appendix F
Quick Reference

The following pages contain reference tables of keyboard and mouse button actions.

Function Keys

KEY		⇑Shift	Alt or Ctrl
F1	Delete To Scrap	Delete, Not to Scrap	NO ACTION
F2	Insert from Scrap	NO ACTION	NO ACTION
F3	Quit & Save All Files	Quit & Discard All Edits	Quit & Ask About Files
F4	Duplicate text	Copy to Scrap	NO ACTION
F5	Extract text	NO ACTION	NO ACTION
F6	Search for Selection	Search for String	Search Backwards
F7	Go Back To Last Place	Go To Selection	Go To Line Number
F8	Redo Last Edit	Find Matching Bracket	NO ACTION
F9	Undo Last Exit	Undo, Erase History	Go to Selected Line Number
F10	Redraw the Screen	Load Selected File Name	New Window Selected File Name

Alt-Letter Keys

Alt-A	Enter **ASCII** character.
Alt-B	Bottom the active window.
Alt-C	Close the active window.
Alt-D	Execute the *DOS* command interpreter.
Alt-E	Exchange the selection with the scrap.
Alt-G	Goto line number.
Alt-H	Enter the help system.
Alt-I	Display debugging information.
Alt-L	Load new file into the active window.
Alt-M	Record keystrokes in the macro buffer.
Alt-N	Create new window.
Alt-O	Toggle overtype mode and insert mode.
Alt-P	Play back keystrokes from the macro buffer.
Alt-Q	Quit and ask about unsaved files.
Alt-R	Global replace.
Alt-S	Save file in the active window.
Alt-T	Toggle 43 line mode. *IBM EGA only.*
Alt-U	Undo.
Alt-W	Write file in the active window.
Alt-X	Hide the window.
Alt-Z	Zoom or unzoom the active window.

Keypad Cursor Keys

Key	Command
PgUp	Scroll up
PgDn	Scroll down
Ctrl-PgDn	Go to End of file
Ctrl-PgUp	Go to Beginning of file
Home	Move cursor to edge of screen
End	Simulate mouse buttons
↑ ↓ ← →	Move mouse cursor one space in any direction
Ctrl-←	Move mouse cursor one word (blank-delimited) left
Ctrl-→	Move mouse cursor one word (blank-delimited) right
Grey +	Fast cursor motion
Grey −	Move cursor to window edges and corners
Del	Delete selection to scrap
Ins	Insert from scrap
← Back	Delete previous character
Ctrl-← Back	Delete previous word
Esc	Cancel command
Ctrl-Break	Stop current editor action

Mouse Command Chart

Cursor Position	■□□	□■□	□□■
Outside Window	NO ACTION	**TOPLIST** menu	NO ACTION
Inside Window	Select text	**TOPLIST** menu	Extend selection
Any Corner	Toggle window top/bottom	NO ACTION	Stretch window
Top Border	Activate window	Split window (vertical)	Move Window
Right Border	NO ACTION	Split window (horizontal)	NO ACTION
Left Border	Scroll Up	Thumb (by line)	Scroll Down
Bottom Border	Scroll Left	Thumb (by column)	Scroll Right

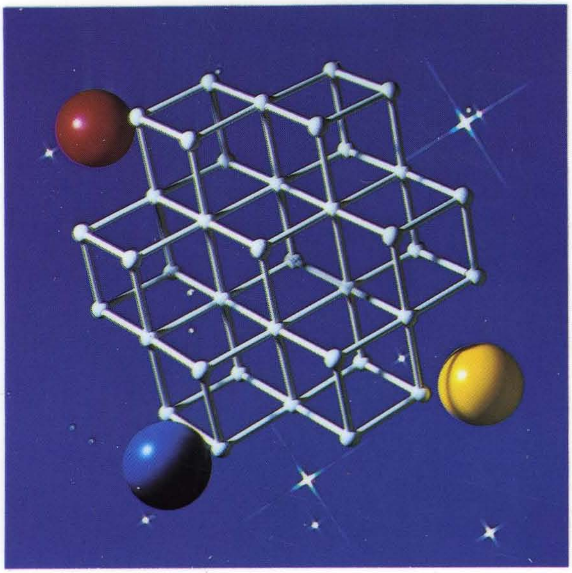
Notes:

INDEX

Notes:

Notes:

LOGITECH™ MODULA-2 VERSION 3.0



POINT EDITOR

LOGITECH

Logitech U.S.A.
Corporate Headquarters
6505 Kaiser Drive
Fremont, CA 94555
Tel: 415-795-8500

Logitech Switzerland
European Headquarters
CH-1111 Romanel/Morges
Switzerland
Tel: 41-21-869-9656

Logitech Taiwan
Far East Headquarters
15 R&D Road 2
Science Based Industrial Park
Hsinchu, Taiwan, ROC
Tel: 886-35-77-8241

Algol-Logitech Italy
Via Durazzo 2
20134 Milano MI
Italy
Tel: 39-2-215-5622