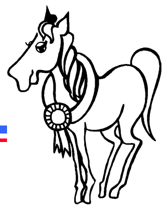
Qedit Cookbook for Novices



A Robelle Tutorial

Bob Green, August 1995

robelle
solutions technology

1

Qedit is Robelle's full-screen editor for programmers on MPE and HP-UX. Many users have Qedit on their system, but have never seen the manual or attended a Qedit training course. This tutorial is for those who use Qedit every day, but only know the few features shown to them by co-workers.

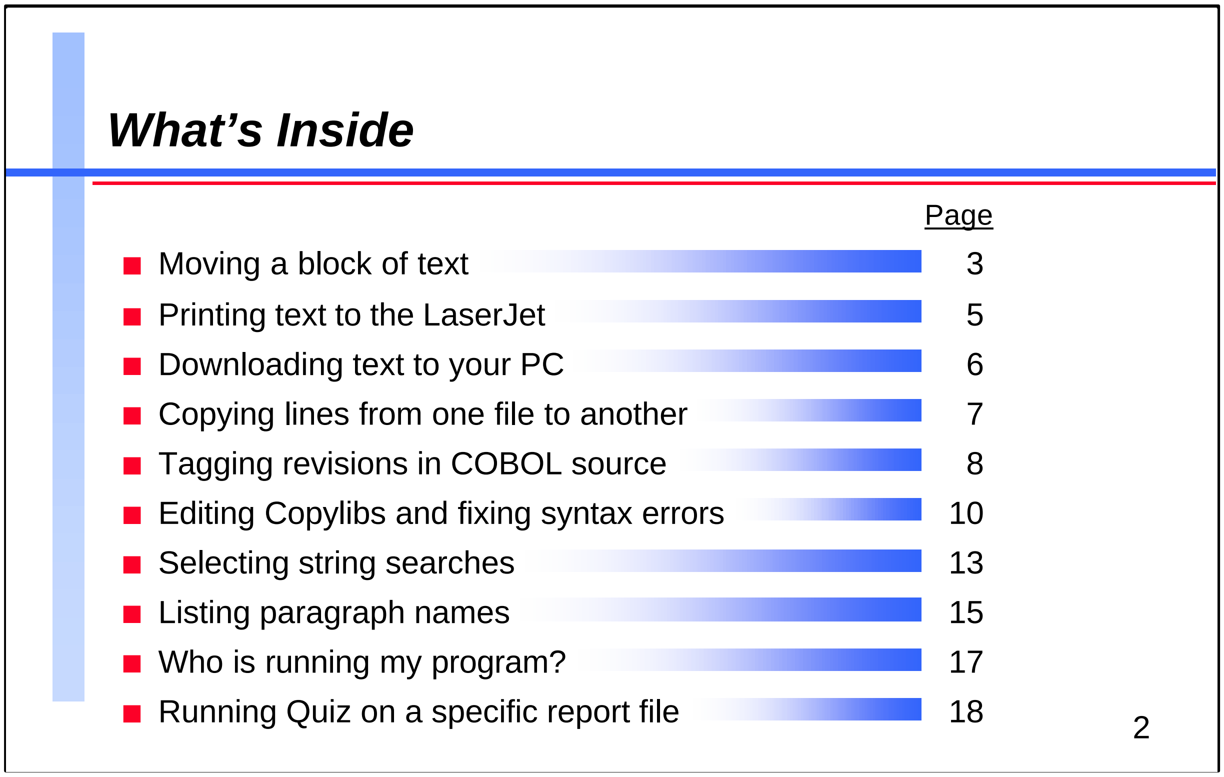
You will learn how Qedit can help you solve common programming problems so you can use Qedit more effectively in your work. This tutorial will present real examples with actual input data and output results, then highlight the Qedit features used to obtain these results. For example, we will examine how to tag COBOL source revisions, as well as how to conveniently write and test Quiz reports. You will also learn to use the Hold file to copy lines between files and how to add new user commands to Qedit such as a smart Dir command that can list files on both MPE and DOS.

Qedit Cookbook for Novices is written by Bob Green, founder of Robelle. Bob is the programmer in charge of Qedit, and the best person to ask about enhancements.

Robelle Solutions Technology Inc.
Suite 201, 15399 - 102A Avenue
Surrey, B.C. Canada V3R 7K1
Toll-free: 1.888.762.3553
Telephone: 604.582.1700
Fax: 604.582.1799
E-mail: support@robelle.com

For the Techies

References
For further information on topics covered in this tutorial, please consult the *Qedit User Manual*.



What's Inside

	Page
Moving a block of text	3
Printing text to the LaserJet	5
Downloading text to your PC	6
Copying lines from one file to another	7
Tagging revisions in COBOL source	8
Editing Copylibs and fixing syntax errors	10
Selecting string searches	13
Listing paragraph names	15
Who is running my program?	17
Running Quiz on a specific report file	18

2

This tutorial will teach you how to use Qedit to solve a number of common, non-trivial problems.

For instance, in the first example we locate a specific block of text, mark it, and then move it to another location in the same file. To find out more about the Qedit commands used in this task (Find String, Forward Page, Update, MM, A), please refer to the *Qedit User Manual*.

We will end the tutorial with an exercise that challenges you to use what you have learned.

For the Techies

References

Moving a block of text

■ Open file for edit:	TEXT or OPEN command
■ Mark block:	MM in screen mode
■ Search file:	F6 for next page, F5 for previous page
■ Move marked block:	A to insert after current line
■ Save file:	KEEP or SHUT command

3

Qedit is a full-screen editor for programmers. Suppose your COBOL program has a declaration in the wrong place and you need to move it to the correct location. You can use Qedit to perform this simple task. First you need to bring the file into Qedit, then you can edit the file and save the changes.

If you are using standard format editor files for your source code, use the Text command to copy a file into Qedit and the Keep command to save it. If you have converted your source files into Qedit format files, use the Open and Shut commands instead. Qedit files have a code of 111 and store text in a compressed format that can also be edited directly.

Convert a file to Qedit format:	`text filename;shut *; open *`
Get into Screen mode:	press the F1 key or type `vis`
Get back to Line mode:	press the F8 key

For the Techies

References

Marking a block of text before moving it

- Place MM on the first and last line, then press Enter

```
===>
Okay  54  PROG1.SRC
-3     05 item-description   pic x(50).
-2
-1MM FD line-printer
*       data record is line-record.
+1  01 line-record          pic x(132).
+2MM
//  ....+....1....+....2....+....3....+
```

- To insert text after a specific line, mark an A on it and then press Enter

4

The screen is divided into four parts: the command line (===>), the status line, the text area, and the template line. Within the text area, the current line is marked with an asterisk (*). Lines above the current line are marked -1, -2, -3 and lines below are marked +1, +2, +3. Qedit does not show the actual line numbers since they are usually not needed.

Adjust screen size: `set vis above 3 below 15`

Other cut and paste functions:

`CC`	copy
`DD`	delete
`B`	move/copy before this line

Repeatedly insert the same text: Cut and paste operations save lines in a temporary file called Hold0. To copy text from the Hold0 file, use these special functions:

`A0`	insert after this line
`B0`	insert before this line

For the Techies

On HP-UX, Hold0 is actually called /usr/tmp/qholdBAAa25295.0, but Qedit will find it if you use **`list hold0`**. Since UNIX does not have temporary files, it generates the random filename BAAa25295.0.

References

For a full list of cut and paste operators, type **?** in the command line (also the home up position on your screen) and press Enter.

Printing a block of text on the LaserJet in a two-up format

- Use ZZ to mark a block, LIST in the command line, then press Enter

```
===>list $lp $pcl 10 zz    {PCL 10 prints two-up, landscape}
Okay  54  PROG1.SRC
-3    05 item-description   pic x(50).
-2
-1ZZ FD line-printer
*       data record is line-record.
+1   01 line-record         pic x(132).
+2ZZ
//   ....+....1....+....2....+....3....+
```

5

One reason you don't need to see line numbers is the **magic marker** feature. Using the ZZ command, you can mark any block of text and then execute commands on these lines.

```
    ===>change "line-record"linebuf-rec"zz
    ===>justify format margin 65 zz  {format into paragraph}
    ===>lq $record $shift zz         {attached printer}
```

Other PCL codes (add 1000 for ASCII, 2000 for A4 paper):

PCL 1 - landscape, tiny lineprinter font
PCL 2 - landscape, 100 columns, Courier font
PCL 3 - portrait, 80 columns, Courier font
PCL 4 - portrait, tiny lineprinter font
PCL 5 - portrait, Courier tightened for A4 paper

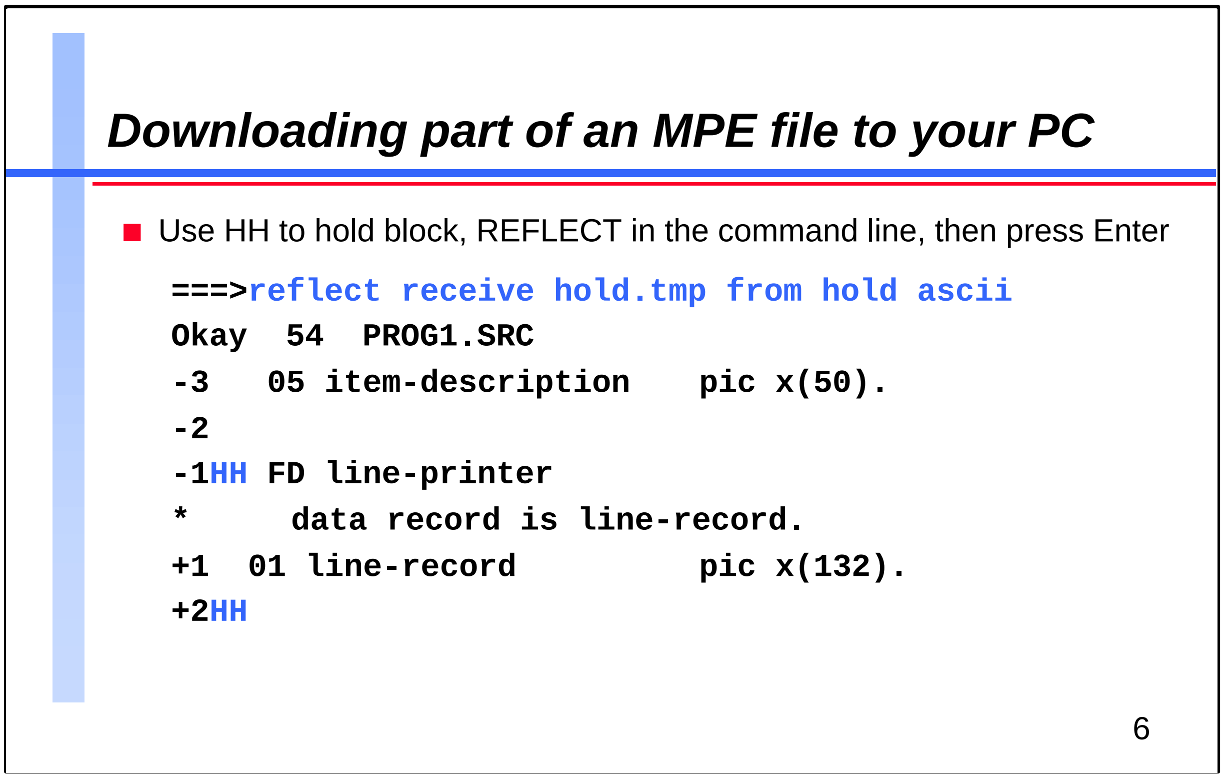
Other List options:

`$double`	double spaced
`$duplex`	two-sided listing on LaserJet (not HP-UX)
`$page off`	without page headings
`$include`	also search $include files
`$char`	suppress control characters
`$hex`	show hex values for characters

For the Techies

Note that List to $lp and $record are not supported in Qedit/UX.

References



Downloading part of an MPE file to your PC

- Use HH to hold block, REFLECT in the command line, then press Enter

```
===>reflect receive hold.tmp from hold ascii
Okay  54  PROG1.SRC
-3   05 item-description    pic x(50).
-2
-1HH FD line-printer
*      data record is line-record.
+1  01 line-record          pic x(132).
+2HH
```

6

The HH cut and paste function copies a block of lines into a temporary file called Hold.

```
:listftemp hold,1
TEMPORARY FILES FOR BOB.USER,BOB
ACCOUNT=  USER    GROUP=  BOB
FILENAME CODE -------LOGICAL RECORD------
               SIZE    TYP     EOF   LIMIT
HOLD   *       1000B    FA       4   250000 (TEMP)
```

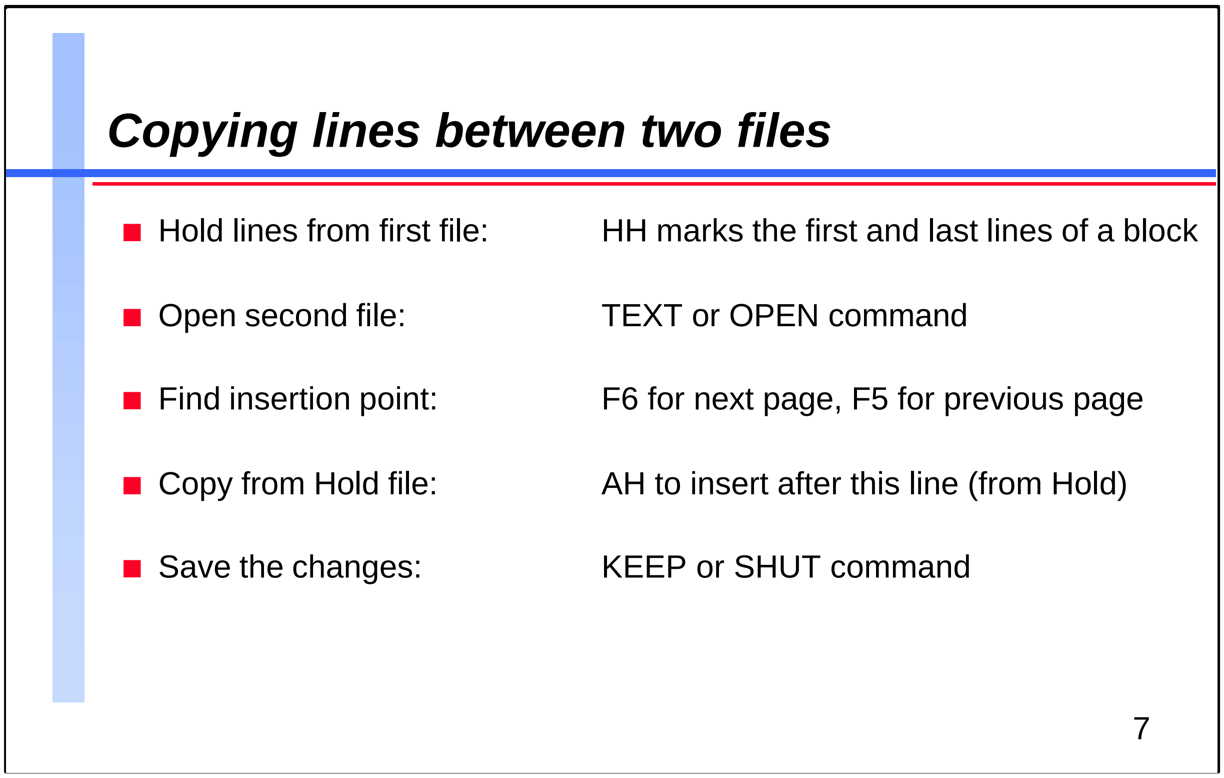
Reflection: If you are using a Reflection terminal emulator on your PC, you can use Qedit's :Reflect command to execute PC functions under the control of your HP 3000.

Upload a file:	`reflect send host.tmp to host ascii`
Copy from Hold:	AH to insert after this line
	BH to insert before this line

For the Techies

The :Reflect command is not supported on HP-UX.

References



Copying lines between two files

■ Hold lines from first file:	HH marks the first and last lines of a block
■ Open second file:	TEXT or OPEN command
■ Find insertion point:	F6 for next page, F5 for previous page
■ Copy from Hold file:	AH to insert after this line (from Hold)
■ Save the changes:	KEEP or SHUT command

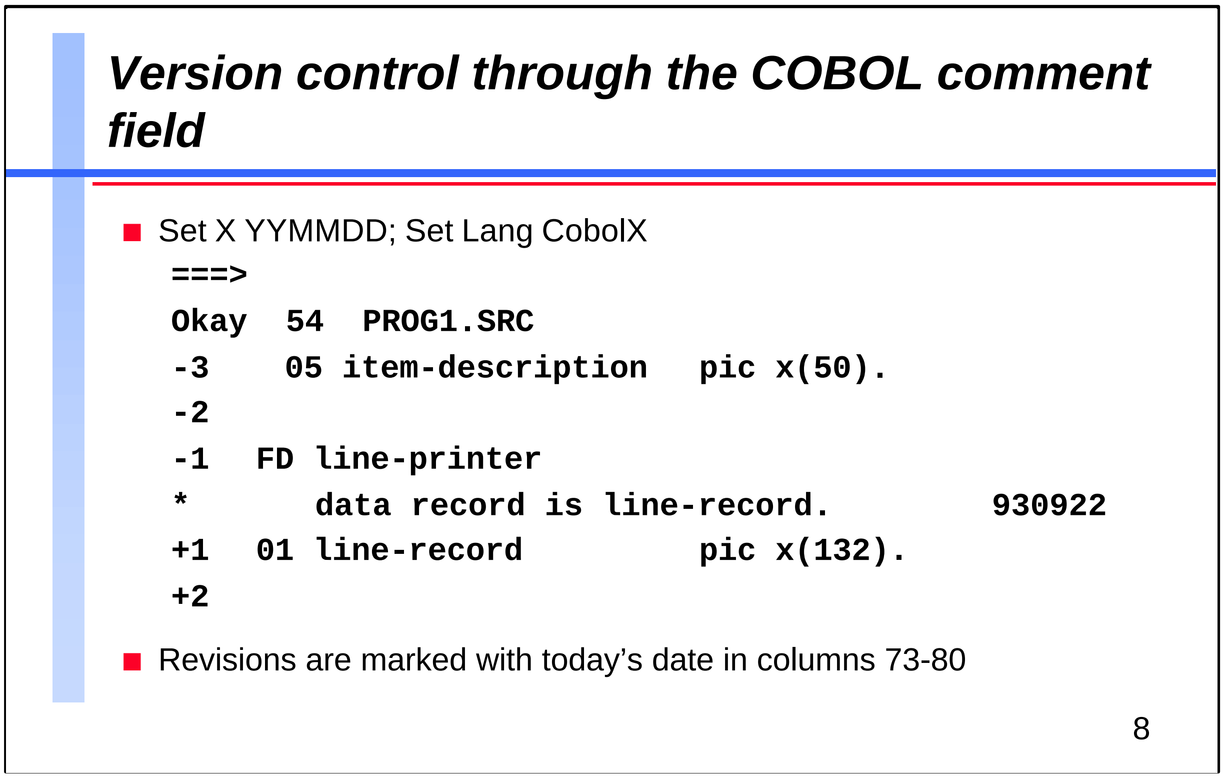
7

You can combine the basic edit functions you have just learned in hundreds of ways to solve many different problems.

For example, this slide shows you how to copy text from one file to another.

For the Techies

References



Version control through the COBOL comment field

- Set X YYMMDD; Set Lang CobolX

```
===>
Okay   54   PROG1.SRC
-3     05 item-description   pic x(50).
-2
-1    FD line-printer
*        data record is line-record.        930922
+1    01 line-record         pic x(132).
+2
```

- Revisions are marked with today’s date in columns 73-80

8

Qedit offers a special feature for COBOL users called **change tagging**. After you enable the Set X command, any further modifications to a COBOL program will be tagged with the current date. Qedit inserts the date in columns 73-80 and lets you see exactly when specific lines were changed. The Set X command is activated only if you are working on a COBOL source file and only after Set Lang CobolX is enabled.

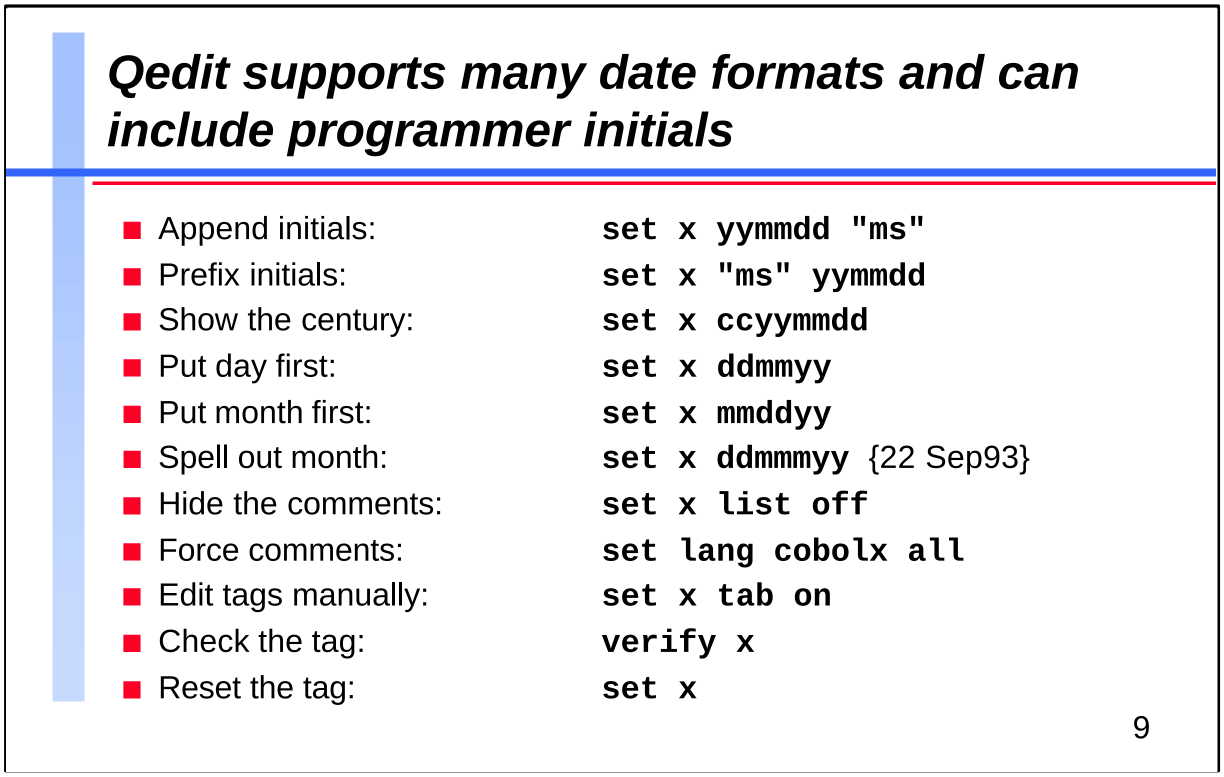
Qedit recognizes two formats for COBOL source: programs with comment fields (called COBOLX for "eXtended") and programs without comments. Most people omit the comment fields when writing a new program to avoid compile time errors caused by source code inadvertently extending past column 72. To select a format without comments, you can use `set lang cobol`.

If you want to start documenting revisions once a program goes into production, use `set lang cobolx` to add comment fields. Then you can enable the Set X command to ensure that all subsequent changes to the program are carefully tagged with a date in columns 73-80.

Configure Set X every time: put Set X command into Qeditmgr file

For the Techies

References



Qedit supports many date formats and can include programmer initials

Append initials:	`set x yymmdd "ms"`
Prefix initials:	`set x "ms" yymmdd`
Show the century:	`set x ccyymmdd`
Put day first:	`set x ddmmyy`
Put month first:	`set x mmddyy`
Spell out month:	`set x ddmmmyy` {22 Sep93}
Hide the comments:	`set x list off`
Force comments:	`set lang cobolx all`
Edit tags manually:	`set x tab on`
Check the tag:	`verify x`
Reset the tag:	`set x`

9

The Set X command tags all the changed lines in COBOLX files with a string and/or the current date. As you can see above, Qedit supports a wide variety of date formats. Whatever revision tag you select, it appears in columns 73-80 of lines modified or added.

To hide the comments during normal text editing or listing, use **`set x list off`**. You can ensure that all COBOL source files are tagged with comments by using **`set language cobolx all`**. This command automatically converts COBOL to COBOLX.

In Visual mode, the COBOLX margin is usually set at column 72 to prevent comments from shifting left or right when you make changes to your code. If you have to maintain the comment field manually, this margin setting means you must use the cursor instead of a tab to move to column 72. In order to move to the comment field with a tab, you have to reset the margin from column 72 to column 80 by using **`set x tab on`**.

Check the current tag value:	**`verify x`**
Reset the tag:	**`set x ""`**

For the Techies

You can enable the Set X command only if the program has a full record length (80 columns).

References

COBOL Copylibs on MPE

- Problem: How can I edit a Copylib member?
- Answer: Put the member name in parentheses.

```
/:file copylib=copylib.cobsrc

/text (custrec)

/change "customer"Customer" all

/keep
```

10

COBOL lets you maintain Copylibs or libraries of common source code and data. Each library file consists of several members and each member name can be up to eight characters. You can use HP's Cobedit or Ksamutil to create the original Copylib file and then use Qedit commands (List, Text, Keep and Destroy) to maintain it.

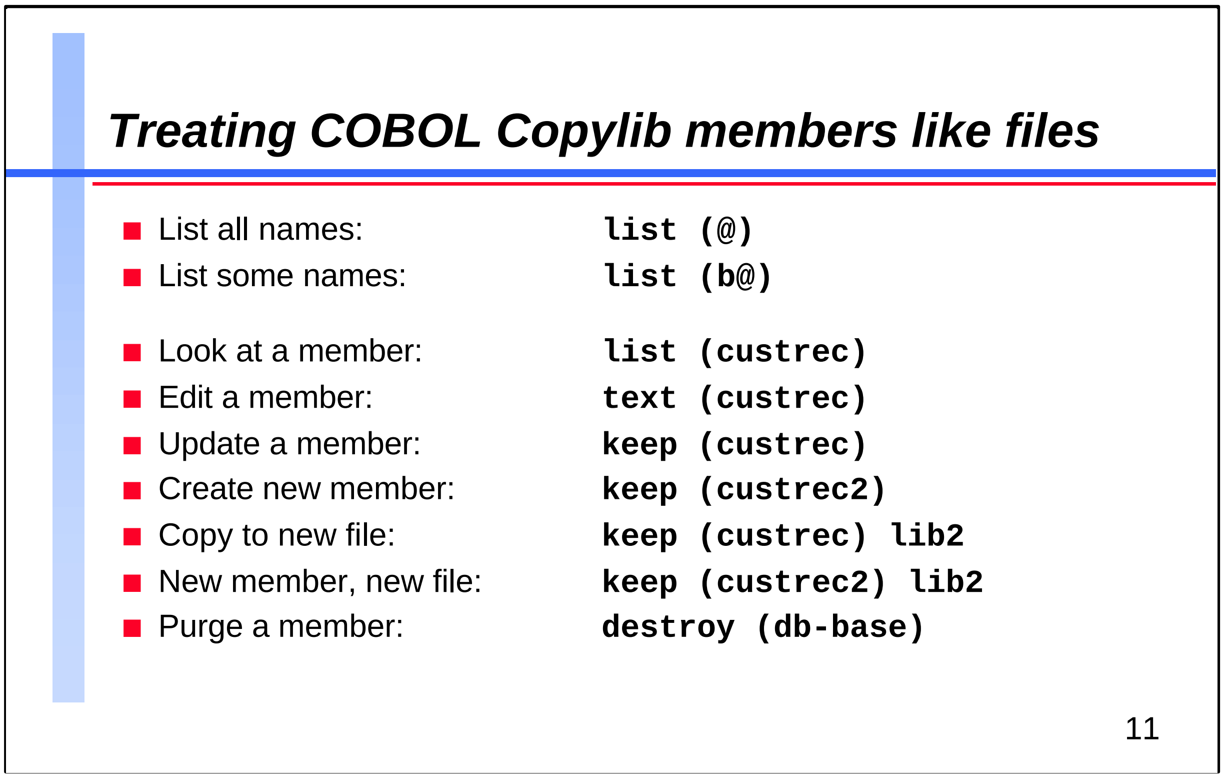
In order to list or edit a Copylib member, you just need to point a :File command to the Copylib file and then put the member name in parentheses as you would a filename. The first step is almost automatic because most COBOL programmers already have a :File equation for their Copylib file. In the second step, you need to enclose the member name in parentheses before you can use Qedit commands (List, Text, Keep etc.). Since the Keep command defaults to the last name used in the Text command, you don't even have to type the member name when saving your changes.

Remember that you need write access to the Copylib file in order to change it.

For the Techies

Qedit/iX supports NM and CM KSAM Copylib files.

References



Treating COBOL Copylib members like files

■ List all names:	`list (@)`
■ List some names:	`list (b@)`
■ Look at a member:	`list (custrec)`
■ Edit a member:	`text (custrec)`
■ Update a member:	`keep (custrec)`
■ Create new member:	`keep (custrec2)`
■ Copy to new file:	`keep (custrec) lib2`
■ New member, new file:	`keep (custrec2) lib2`
■ Purge a member:	`destroy (db-base)`

11

You can use pattern-matching characters in parentheses to list the members in your Copylib. For example, use **`list (@)`** to print all the member names, or use **`list (b@)`** to list only the names that start with the letter b. Since COBOL compilers require a Copylib file, Qedit looks for the default Copylib through a pointer in the :File equation.

To refer to a Copylib member, put the member name in parentheses and Qedit treats it as a filename.

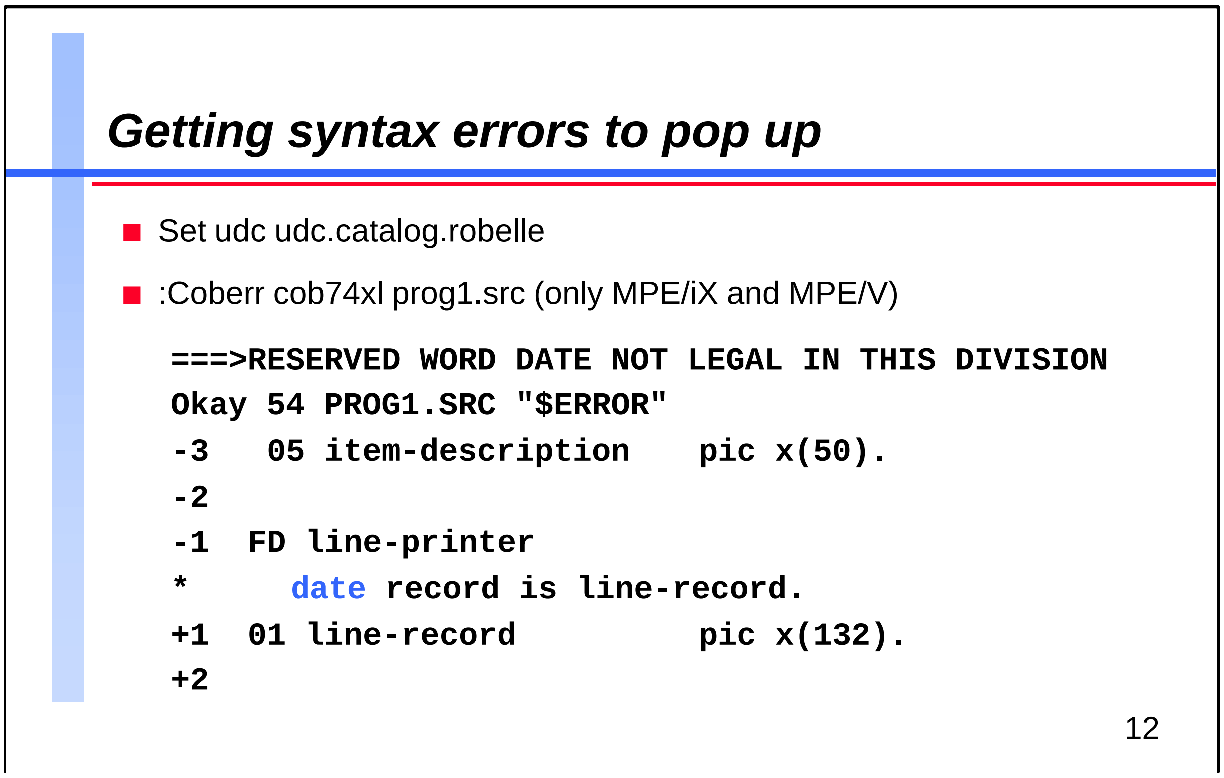
```
                         list (o2314buf) "order-status"
                         list $lp (02314buf)
```

To refer to other Copylib files, you can state them explicitly instead of changing your :File command.

```
                         list (dbparms)
copylib.cobsrc.newsys
                         list $lp (@)
copylib.cobsrc.newsys
```

For the Techies

References



Getting syntax errors to pop up

- Set udc udc.catalog.robelle
- :Coberr cob74xl prog1.src (only MPE/iX and MPE/V)

```
===>RESERVED WORD DATE NOT LEGAL IN THIS DIVISION
Okay 54 PROG1.SRC "$ERROR"
-3   05 item-description    pic x(50).
-2
-1  FD line-printer
*      date record is line-record.
+1  01 line-record          pic x(132).
+2
```

12

If you would like Qedit to locate errors in your COBOL source files and display error messages, try the :Coberr User Defined Command (UDC). To enable the :Coberr command, use **`set udc udc.catalog.robelle`**. Instead of using a regular compile command, the :Coberr UDC specifies an alternate compile command as the first parameter and a source file as the second parameter.

```
:coberr cob74x1,myprog.source
:coberr cob85x1,*   {current file}
```

You use the same syntax as in compiling, but you insert :Coberr in front of the command. For example, instead of typing **`cob85x1 *`**, you type **`:coberr cob85x1,*`**. The default compile option is Cob85xl, but you can modify your UDC to make it consistent with your own shop. The :Coberr command can also have parameters for the OBJ/USL output file and for the temporary disc file with the compiler listing.

After compiling your source file, Qedit puts you into Screen mode. It places the cursor on the offending word and displays the COBOL error message in the command line. In this example, Qedit puts the cursor on the word "date" because the correct word is data. You can correct the error and press Enter to continue. When you press F4, Qedit goes to the next compile error.

For the Techies

References

How to find "customer" without getting "customer-record"

- **`list "customer"`**

```
01 customer-record.
   05 customer            pic x(40).
   05 customer-address    pic x(40) occurs 5 times.
   05 customer-phone      pic x(20).
   05 customer-number     pic 9(6).
```

  5 lines found

- **`list "customer" (smart)`**

```
   05 customer pic x(40).
```

  1 line found

13

Qedit lets you search for strings within any command (List, Delete, Keep etc.). When you perform a regular string search, you may get a lot of unwanted matches because it includes occurrences where the selected string is embedded in another string. See the "customer" example above.

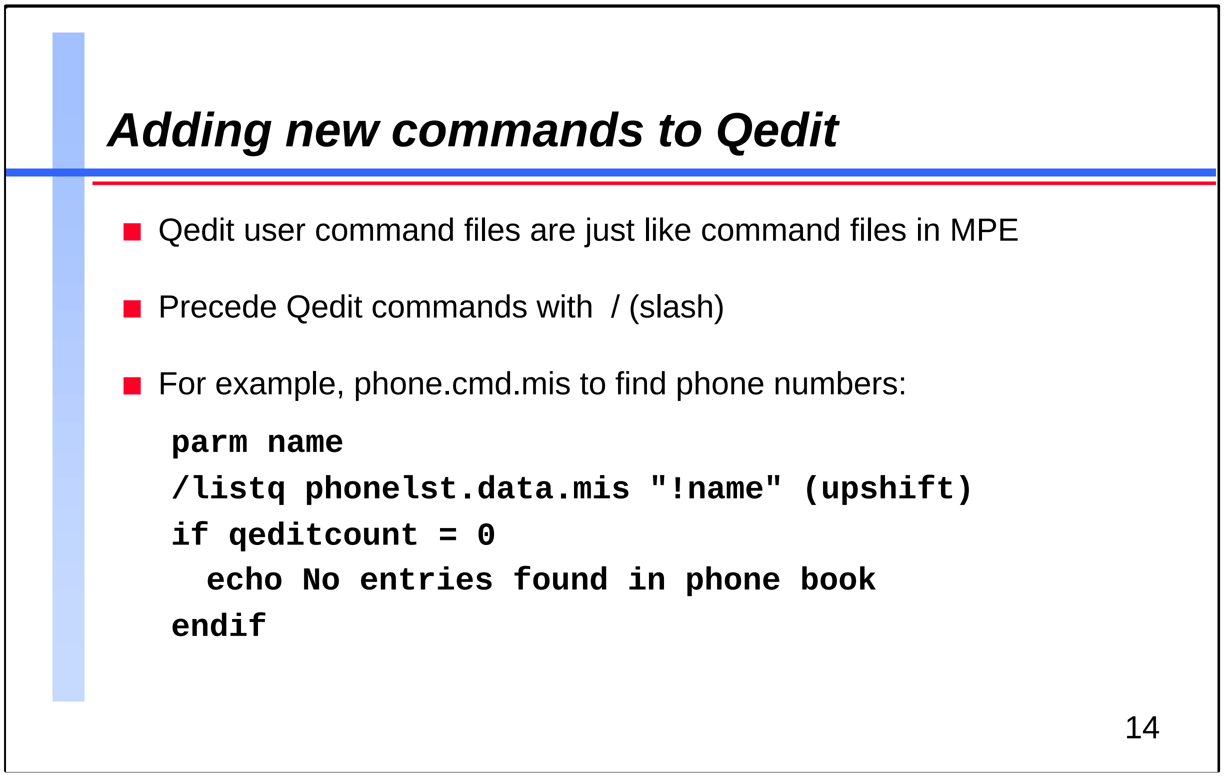
Qedit has a Smart option to get around this problem. When you append (S) or (Smart) to a string search, Qedit will only include occurrences where the selected string appears on its own. With the Smart option, Qedit ignores "customer-record" when looking for "customer" because it knows that a hyphen is valid in a COBOL name.

Smart is called a string window option. There are other useful window options.

Ignore upper versus lower case:	**`list "customer" (upshift)`**
Search a column slice only:	**`list "customer" (5/15)`**
Find lines that don't match:	**`list "customer" (nomatch)`**
Find lines ending in "bob":	**`list "@bob" (pattern upshift)`**

For the Techies

References



Adding new commands to Qedit

- Qedit user command files are just like command files in MPE
- Precede Qedit commands with / (slash)
- For example, phone.cmd.mis to find phone numbers:

```
parm name
/listq phonelst.data.mis "!name" (upshift)
if qeditcount = 0
  echo No entries found in phone book
endif
```

14

You can add new commands to Qedit by creating user command files that contain MPE commands and parameter definitions. Qedit command files are like those in MPE, except they can also execute Qedit commands (preceded with /).

You can run a user command file by typing its filename followed by any parameters. If the command file resides on your HPPATH list of groups, you don't need to type the full name.

Invoke a command file:	`/phone ralph`
String search with parameters:	`/list file "!name"`
Test Qedit results:	`if qeditcount = 0` {automatic JCW}

For the Techies

Shell scripts can be executed in Qedit/UX, but Qedit/UX does not support command files.

References

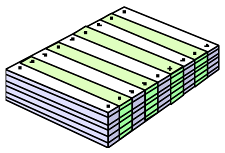
How can I list all my COBOL paragraph names?

- Use FIND to get to the start of the Procedure Division

```
/findq "Procedure Division" (upshift) first
```

- Look for a blank in column 7 and an alphanumeric in column 8

```
/listq " ?" (7/8 pattern) */last
```



15

This solution assumes that the selected string (Procedure Division) does not occur in comments or in data before the actual Procedure Division. The Listq command searches each line, from the current position to the end of the file, and lists only the lines with a space in column 7 and an alphanumeric character in column 8. The question mark (?) is a pattern-matching character that specifies a single alphanumeric character.

A handy feature to add to the List command is the J (jumping) option. With this option, Qedit only shows you one screen at a time and then prompts you with `More? [yes]:`. You can press Return to see the next screen or enter a command to stop the listing.

```
/lqj " ?" (7/8 pattern) */last
```

The pattern-matching characters:

@	zero or more characters of any type
#	exactly one numeric character (0 to 9)
?	exactly one alphanumeric character
~	zero or more blank characters
&	escape character, next character is literal match
^ and !	reserved for future use

For the Techies

References

Creating a command file to list the Procedure Division

```
setjcw cierror 0
/zz */*            {save the current line number}
continue
/findq "procedure division" (8/25 ups) first
if cierror <> 0 then
    echo Warning: No Procedure Division found
    /listq first
endif
/lqj " ?" (7/8 pattern) */last
/list zz           {go back to where we were}
```

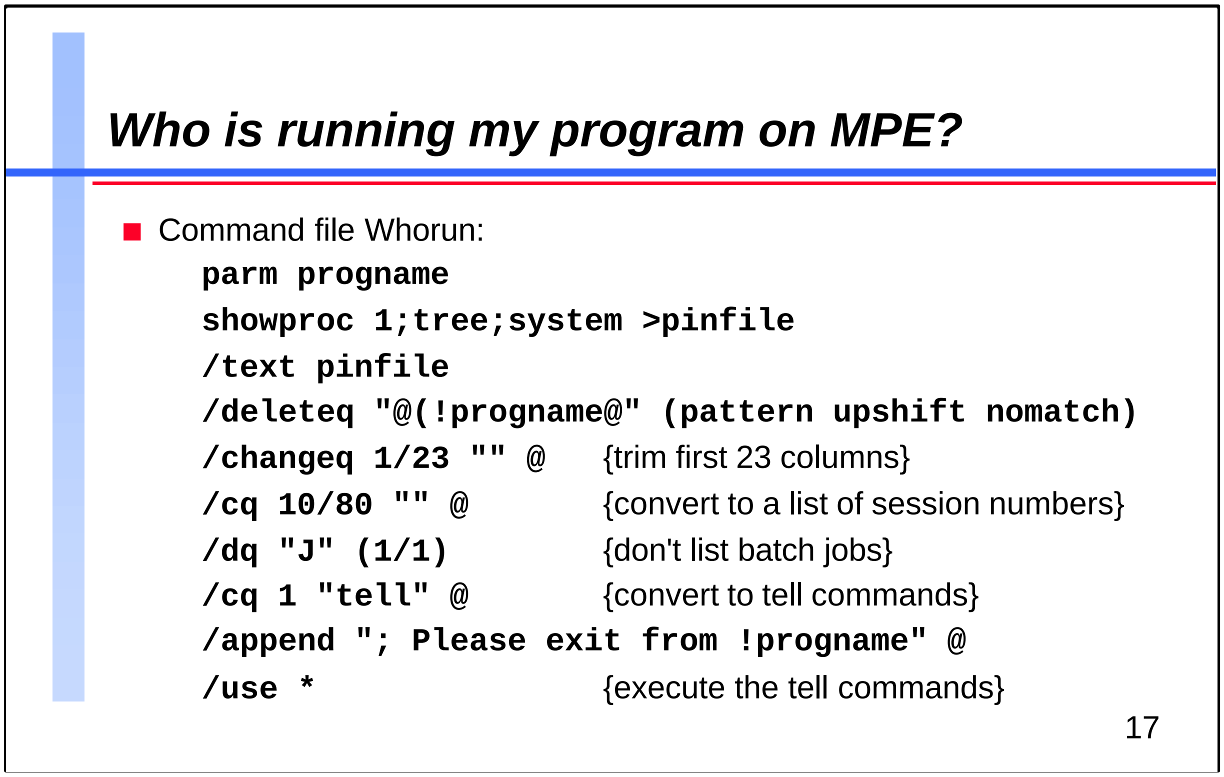
16

In this example, the user command file lists all the procedures in your COBOL source file. You can execute this command file in either Qedit/V or Qedit/iX.

Mark current line:	`/zz */*`
Go on even if Find fails:	`continue`
Jump to marked line:	`/list zz`

For the Techies

References



Who is running my program on MPE?

- Command file Whorun:

```
parm progname
showproc 1;tree;system >pinfile
/text pinfile
/deleteq "@(!progname@" (pattern upshift nomatch)
/changeq 1/23 "" @     {trim first 23 columns}
/cq 10/80 "" @         {convert to a list of session numbers}
/dq "J" (1/1)          {don't list batch jobs}
/cq 1 "tell" @         {convert to tell commands}
/append "; Please exit from !progname" @
/use *                 {execute the tell commands}
```

17

If you don’t have the luxury of MPEX’s :Listf,access command, have you ever wondered who is using a program when you are trying to install a new version? The :Showproc command can show you all the current processes on the system if you have SM capability. To ask all the Qedit users to exit the program, type **`whorun Qedit`**.

This is a listing of Qedit lines produced by the :Showproc command:

```
C155 0:14.152  READY S2192   86   (QEDIT.PUB.ROBELLE)
C152 0:43.466  WAIT  S2190   149  (QEDIT.PUB.ROBELLE)
D202 1:12.597  WAIT  J10806  121  (QEDIT.PUB.ROBELLE)
D202 0:47.812  WAIT  J10805  91   (QEDIT.PUB.ROBELLE)
```

If you remove the first 23 columns, the listing appears as follows:

```
S2192          86    (QEDIT.PUB.ROBELLE)
S2190          149   (QEDIT.PUB.ROBELLE)
J10806         121   (QEDIT.PUB.ROBELLE)
J10805         91    (QEDIT.PUB.ROBELLE)
```

This is a listing of the non-batch processes converted to :Tell commands:

```
tell S2192; Please exit from Qedit
tell S2190; Please exit from Qedit
```

For the Techies

References

Running Quiz on a specific source file

- QZ.CMD.USER {definition}

```
parm auto=$null
anyparm otherparms=ZZZ
file qsource=!auto
if "!otherparms"="ZZZ" then
  quiz "auto=qsource suspend"
else
  quiz "auto=qsource !otherparms suspend"
endif
file qsource=$null
```

18

Since MPE versions of PowerHouse modules can read Qedit format files, you do not have to use the Keep command before PowerHouse works on a file. To access Qedit from within PowerHouse, you can use the Revise command to call up the editor through the COGEDITR :File equation.

Qedit preserves file labels when you use the Text or Keep commands. As a result, you can use Qedit to edit PowerHouse subfiles without damaging the subfile dictionary. Use `text filename,labels` to copy the subfile, and the Keep command to re-create the labels when you save the files.

PowerHouse comes with sophisticated User Defined Commands that allow you to specify almost any Quiz or QTP option. Most of the time, however, you will probably use the same options. In order to minimize typing, it makes sense for you to put frequently used options into a command file.

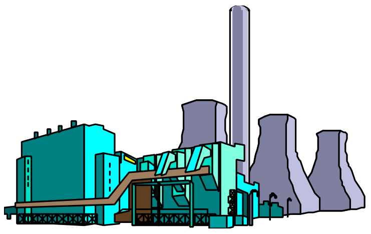
When you invoke Quiz from within Qedit, you may want to select two useful features. The first feature allows you to specify the source filename as a parameter instead of waiting for the prompt to appear. The second feature lets you suspend Quiz when you want to leave it rather than terminating it. If you need to use Quiz later in the same session, this option lets you access it much faster.

For the Techies

References

Examples of invoking PowerHouse command files

- `qz *`
- `qz abc.source`
- `qp * list=no cc=hpxyz`
- `qd *`



You can invoke the QZ command file by specifying the source filename as the first parameter.

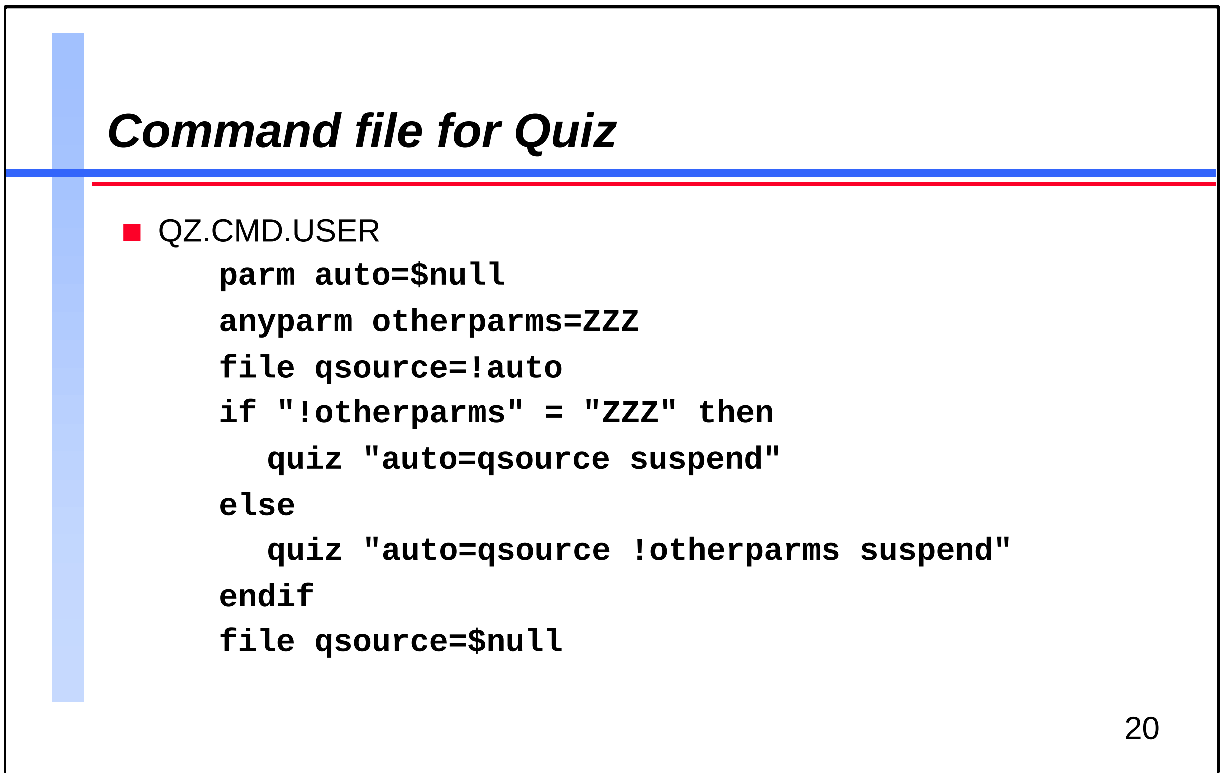
```
/qz srcfile
/qz *          {current file}
/qz $          {last file listed}
```

Notice that when you type **qz ***, Qedit sends your current workfile to Quiz. When an asterisk (*) is specified, Qedit replaces it with the current file. This is a lot faster than typing the entire filename. If the source file has a lockword, Qedit automatically passes it as a parameter with the filename so you don't have to respond to the **Lockword?** question.

The syntax **qx $** invokes the QZ command file and passes the name of the last Qedit file listed or opened.

For the Techies

References



Command file for Quiz

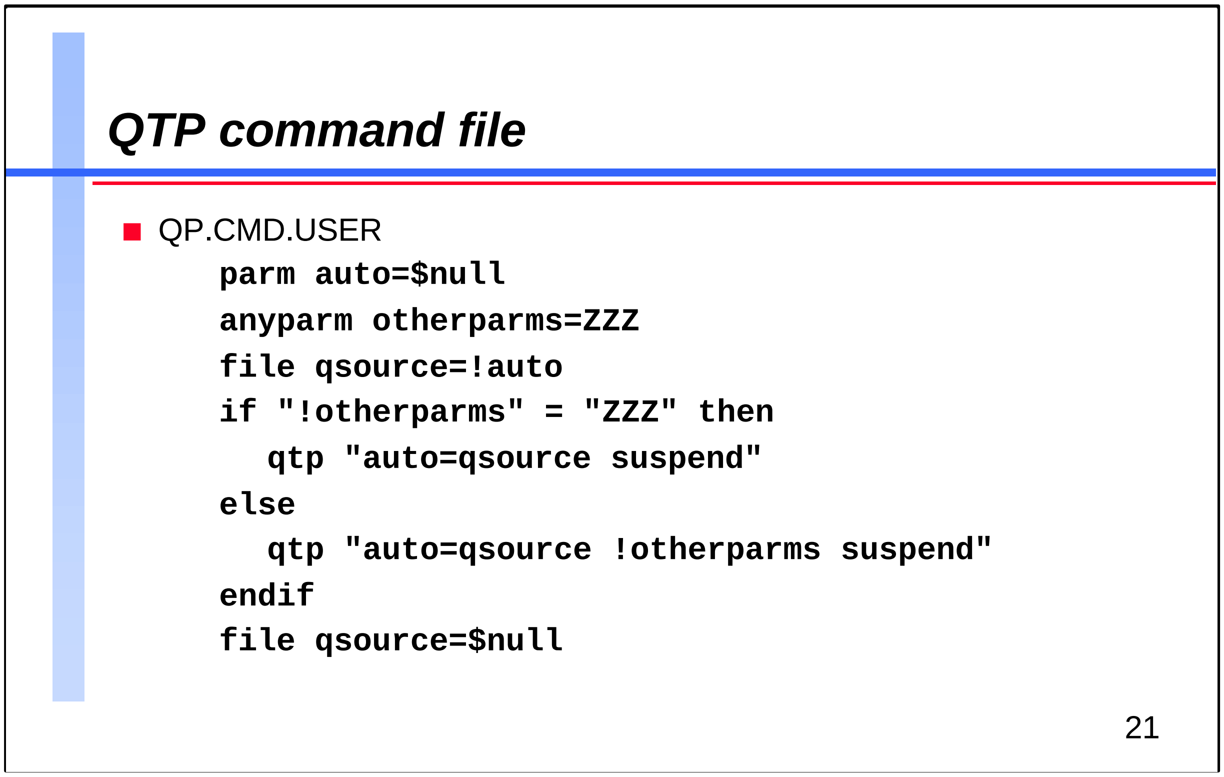
- QZ.CMD.USER

```
parm auto=$null
anyparm otherparms=ZZZ
file qsource=!auto
if "!otherparms" = "ZZZ" then
  quiz "auto=qsource suspend"
else
  quiz "auto=qsource !otherparms suspend"
endif
file qsource=$null
```

20

For the Techies

References



QTP command file

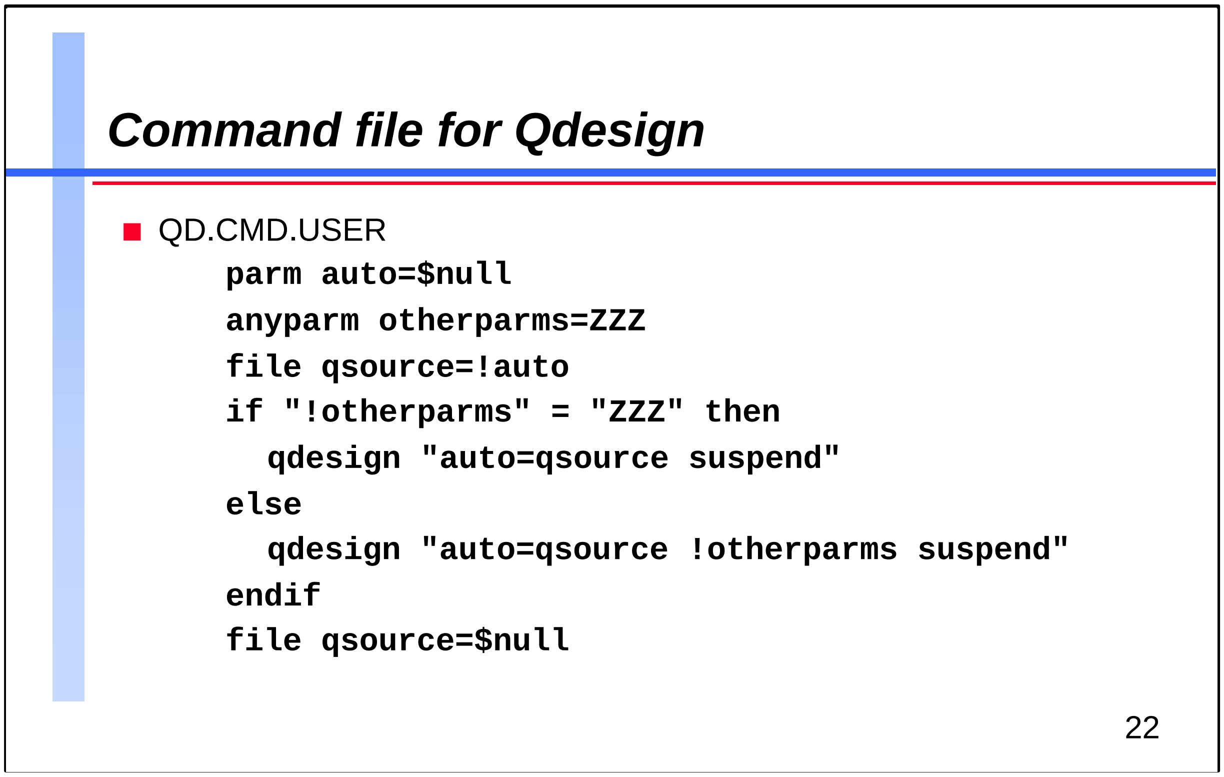
- QP.CMD.USER

```
parm auto=$null
anyparm otherparms=ZZZ
file qsource=!auto
if "!otherparms" = "ZZZ" then
  qtp "auto=qsource suspend"
else
  qtp "auto=qsource !otherparms suspend"
endif
file qsource=$null
```

21

For the Techies

References



Command file for Qdesign

- QD.CMD.USER

```
parm auto=$null
anyparm otherparms=ZZZ
file qsource=!auto
if "!otherparms" = "ZZZ" then
  qdesign "auto=qsource suspend"
else
  qdesign "auto=qsource !otherparms suspend"
endif
file qsource=$null
```

22

For the Techies

References

Qedit Exercise

- Write a DIR command file to list files on your PC or on MPE/iX

`/dir c:\autoexec.bat`	{DOS file}
`/dir q@`	{MPE files starting with Q}
`/dir .\`	{DOS cwd}
`/dir ./`	{MPE cwd}
`/dir \*.bat`	{DOS root directory}

- Use :REFLECT for the PC listings
- Use :LISTFILE for MPE listings

If you are using a Reflection terminal emulator, you can list files on either your MPE or PC system by creating a Qedit command file named Dir. This command file should have the following parameters:

```
parm fileset=
anyparm options=0
```

If Fileset starts with "\" or ".\" or a drive letter such as "c:", assume it refers to PC files. Otherwise, you can use **`:listfile !fileset`**. Instead of :Listf, it is better to use the :Listfile command because it also shows files (source-code, Config_file, /mail/inlist) in the POSIX namespace.

The key to solving this problem is Qedit's :Reflect command.

```
:reflect dir
```

For the Techies

References

Considerations

- Your command file must examine the first and second characters of the Fileset in order to select between DOS and MPE. Why not extract those characters into variables?

```
setvar lfone lft("!fileset",1)
setvar lftwo lft("!fileset",2)
if lfone="\" or lftwo=".\" or &
   alpha("lfone") and pos(":","!fileset")=2
   reflect dir !fileset
```

- We do not pass the !options parameter to the :REFLECT command because Reflection's DIR command does not support options such as /w.

24

For the Techies

References

Solution to Exercise

- This is one possible solution to the extra credit problem:

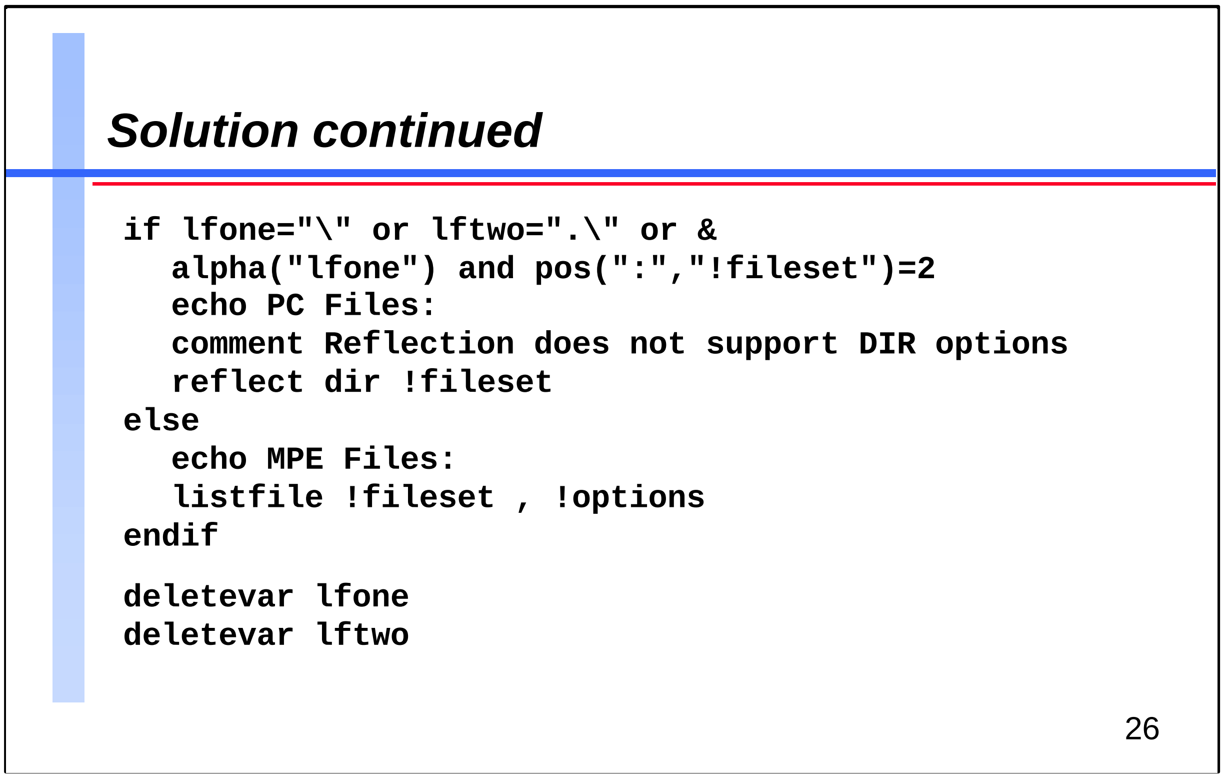
```
parm fileset=
anyparm options=0
comment DIR = list file on DOS or MPE
comment  .\   dos   the current working directory
comment  \    dos   root directory
comment  c:   dos   a specific disk
comment
comment Anything else does a :LISTFILE

setvar lfone lft("!fileset",1)
setvar lftwo lft("!fileset",2)
```

25

For the Techies

References



Solution continued

```
if lfone="\" or lftwo=".\" or &
   alpha("lfone") and pos(":","!fileset")=2
   echo PC Files:
   comment Reflection does not support DIR options
   reflect dir !fileset
else
   echo MPE Files:
   listfile !fileset , !options
endif

deletevar lfone
deletevar lftwo
```

26

For the Techies

References

Summary of Qedit Cookbook

- Screen mode
- LaserJet support
- PC control through Reflection
- COBOL language features
- Search power
- Command files
- PowerHouse convenience
- Mix system functions and Qedit edits

27

We hope you have enjoyed this tutorial.

If you have worked through the entire workbook, you should now be able to do many useful things with Qedit.

For the Techies

References