

## Brutus text editor user guide:

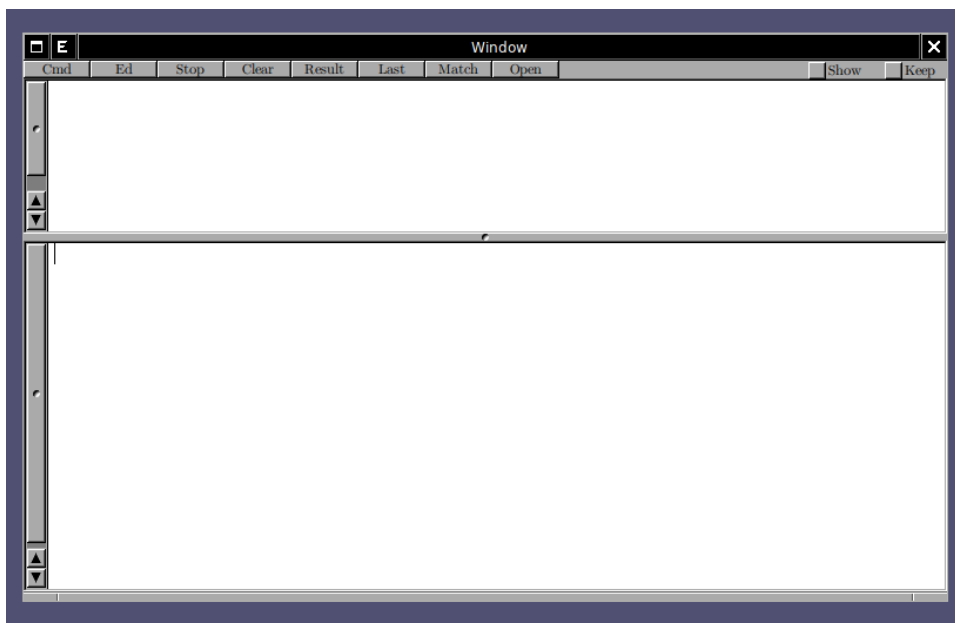
### 1. Why?

Inspired by **Sam**, and **Acme**, from **Plan 9** operating system, my intent is to have an interface that allows system commands, and line the editor **ed**, to edit text. Instead of hard code in the text editor operations such as text substitution, searching, and so on, these operations are delegated to system commands. This gives the user a huge flexibility in its editing operations, and avoid code duplication.

### 2. Why the name?

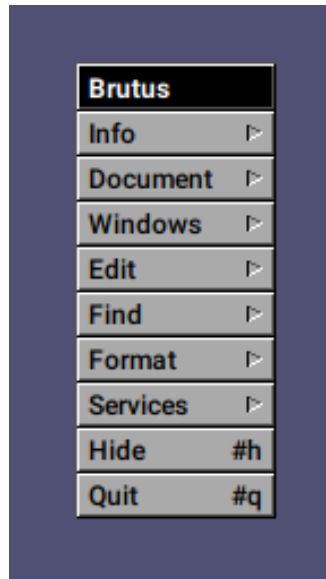
Because it's barebones. There is no syntax, no bells, and it does not do much besides presenting, and organizing text, text files and windows. It's brute.

### 3. The Brutus interface:



The main interface consists of two text areas, and some buttons. The text area on top is the *command area*. This is where commands are inserted. The buttons on the top control, and execute actions based on the content, or text selection, in the command area.

#### 3.1. The Menu:



The Menu allows the user to do the most basic operations, like open files, navigate between the windows, save, etc.

### 3.2. The Cmd button:

This button execute a shell command from the command area.

### 3.3. The Ed button:

This button execute an **ed** command from the command area.

### 3.4. The Stop button:

The Stop button stop the execution of a running command.

### 3.5. The Clear button:

The Clear button wipes the text from the command area.

### 3.6. The Result button:

The Result button place the last result of a command in the command area.

### 3.7. The Last button:

The Last button place the last executed command on the command area.

### 3.8. The Match button:

The Match button match a selection from the command area on the text.

### 3.9. The Open Button:

The Open Button open files from the selected filenames in the command area.

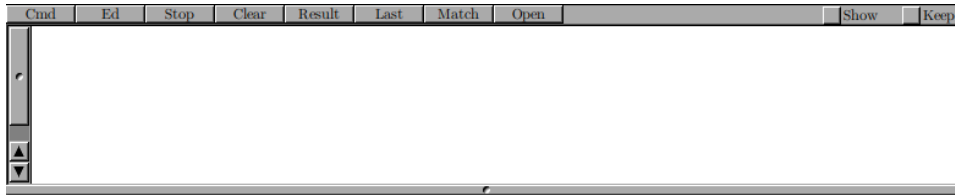
### 3.10. The Show Button:

The Show Button redirects the result of the commands to the command area, instead of applying in the text.

### 3.11. The Keep Button:

The Keep Button prevents the contents of the command area to be erased between commands.

### 4. The command area:



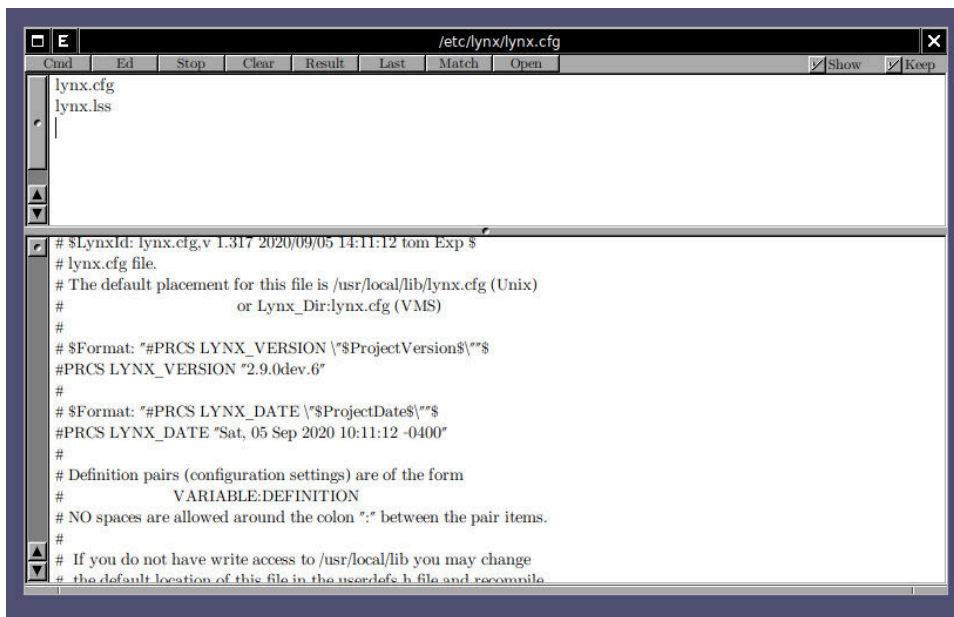
This is where commands are inserted, and executed. Commands can be of two types:

- System commands
- Ed commands

The *Cmd* button execute system commands, and the *Ed* button executes *ed* commands. Examples will be given in the next sections.

### 5. View results of commands:

For example, given that you have a file opened, let's say */etc/lynx/lynx.cfg*. Write in the command area *l s*, check the *Show* button, and click run. You should see this:



### 6. Substitute text using commands:

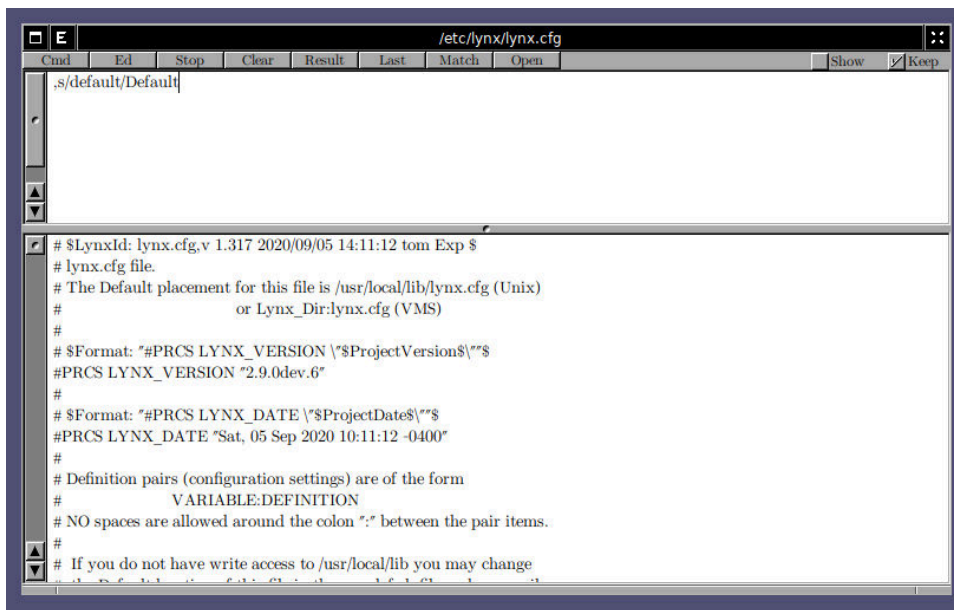
Let's say, in the same file, you want to replace *default* with *Default*. You can use an Ed command like this:

```
1      ,s/default/Default
```

To do that:

- clear the command area pressing the Clear button, if there's text there
- uncheck the Show button, if it's on
- type *,s/default/Default* in the command area
- click the Ed button

You should see this:



```

# $LynxId: lynx.cfg,v 1.317 2020/09/05 14:11:12 tom Exp $
# lynx.cfg file.
# The Default placement for this file is /usr/local/lib/lynx.cfg (Unix)
#           or Lynx_Dir:lynx.cfg (VMS)
#
# $Format: "#PRCS LYNX_VERSION \"$ProjectVersion$\""$
#PRCS LYNX_VERSION "2.9.0dev.6"
#
# $Format: "#PRCS LYNX_DATE \"$ProjectDate$\""$
#PRCS LYNX_DATE "Sat, 05 Sep 2020 10:11:12 -0400"
#
# Definition pairs (configuration settings) are of the form
#           VARIABLE:DEFINITION
# NO spaces are allowed around the colon ":" between the pair items.
#
# If you do not have write access to /usr/local/lib you may change

```

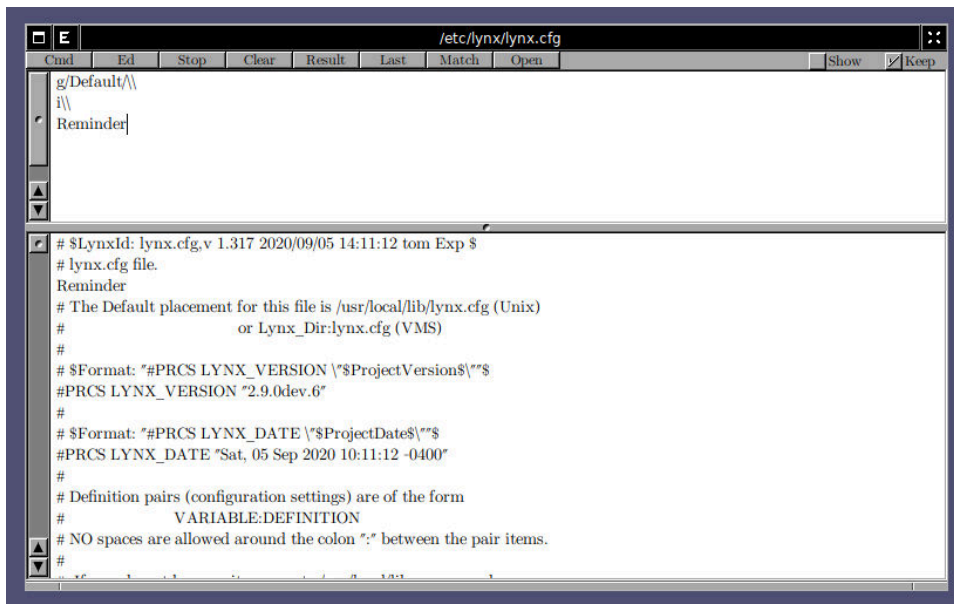
Notice the word default, on line 3, that's now capitalized. This is a trivial example, but you can do more, for example, lets say we want to insert *Reminder* before each line that contained the word Default. We can use another Ed command:

```

1      /g/Default/\
2      i\
3      Reminder

```

You should see this:



```

# $LynxId: lynx.cfg,v 1.317 2020/09/05 14:11:12 tom Exp $
# lynx.cfg file.
Reminder
# The Default placement for this file is /usr/local/lib/lynx.cfg (Unix)
#           or Lynx_Dir:lynx.cfg (VMS)
#
# $Format: "#PRCS LYNX_VERSION \"$ProjectVersion$\""$
#PRCS LYNX_VERSION "2.9.0dev.6"
#
# $Format: "#PRCS LYNX_DATE \"$ProjectDate$\""$
#PRCS LYNX_DATE "Sat, 05 Sep 2020 10:11:12 -0400"
#
# Definition pairs (configuration settings) are of the form
#           VARIABLE:DEFINITION
# NO spaces are allowed around the colon ":" between the pair items.
#

```

## 7. Opening files from results of commands:

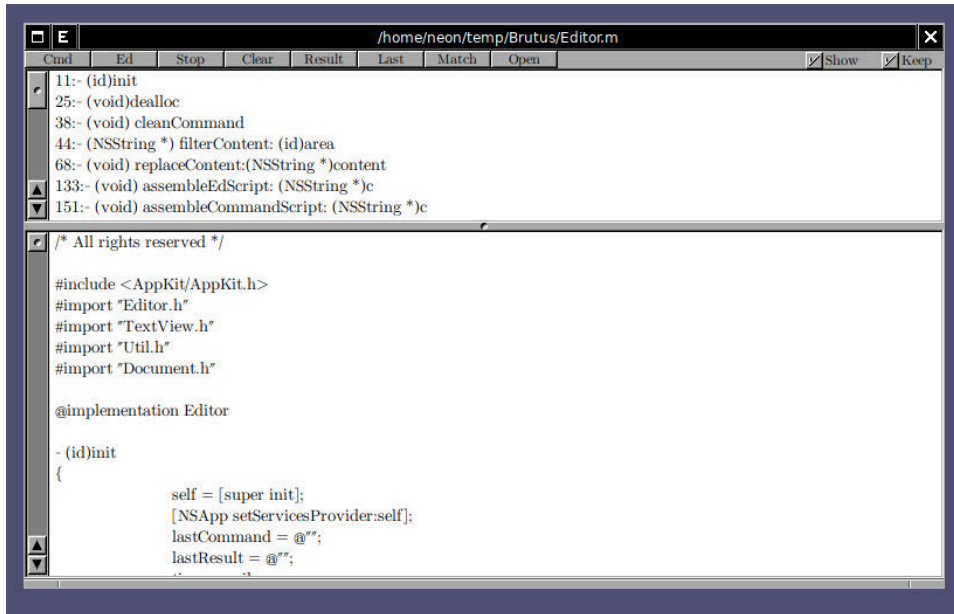
You can highlight filenames in the command area, and click the Open button. This will open the files for editing.

## 8. Match line numbers or strings from results of commands:

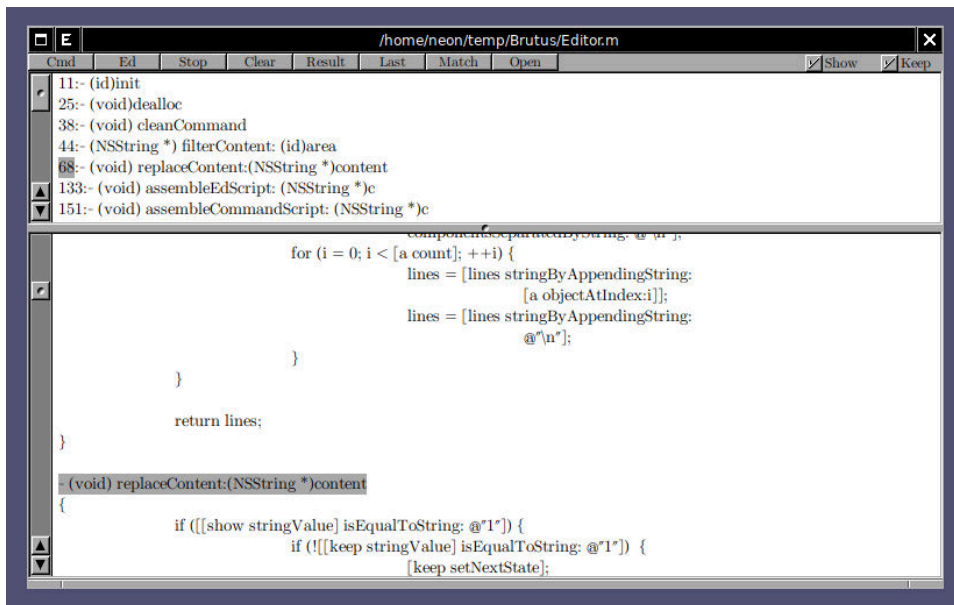
Let's say you are editing an *Objective-C* You can use **grep**, like this:

```
1      grep -n '^[+-]'
```

Type this in the command area, enable Show, and click Cmd. You should see this:



From there, you can highlight a line number (the number before :), or the line contents (the string after :), and click match. You should see this:



Where I selected 68 in the command area, and clicked the Match button.

## 9. The client:

In the directory *BrutusClient*, there is an application that allows you to open files from command line, scripts, etc. After installed, it can be used like the following:

```
1      openapp BrutusClient /path/to/file/name
```

To open the file in a specific line number, let's say, 10, use:

```
1      openapp BrutusClient /path/to/file/name:10
```

To open the file and search for a string, use:

```
1      openapp BrutusClient /path/to/file/name:string
```