ULI101: INTRODUCTION TO UNIX / LINUX AND THE INTERNET

WEEK 2: LESSON 2

MANAGING TEXT FILES:
USING TEXT EDITORS TO CREATE & EDIT A TEXT FILE
MANAGING TEXT FILE CONTENT

LESSON 2 TOPICS

Creating Text Files

- Purpose of a Text Editor
- Using the **`nano`** Text Editor / Demonstration
- Using the **`vi`** Text Editor / Demonstration

Managing / Manipulating Text Files

- Linux Commands: **`touch`**, **`cat`**, **`more/less`**, **`cp`**, **`mv`**, **`rm`**, **`diff`**, **`file`**, **`find`**
- Demonstration

Homework

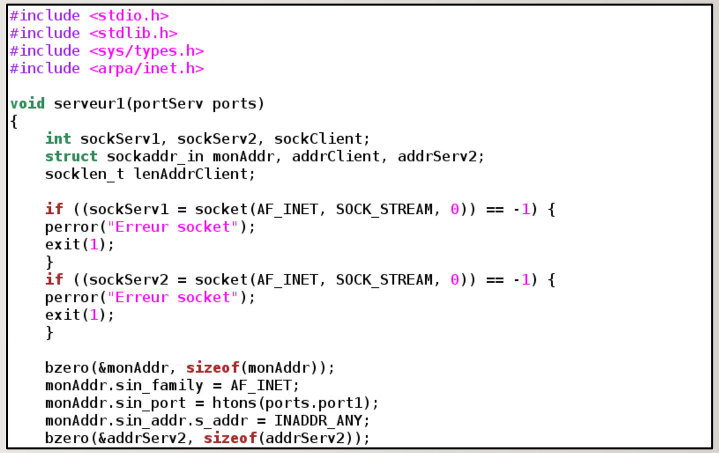
- Perform **Tutorial 2: Unix / Linux File Management (Investigation 2)**
  Perform LINUX PRACTICE QUESTIONS (9 – 16)

CREATING TEXT FILES

Text Editors

A **Text Editor** allows users to **create**, **modify** and **save** editing changes of text files.

Although **programming students** can use **graphical IDE's** to code and compile programs, students can **create source code** using a text editor and **compile their source code** in their Matrix account to generate **executable programs**.



```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <arpa/inet.h>

void serveur1(portServ ports)
{
    int sockServ1, sockServ2, sockClient;
    struct sockaddr_in monAddr, addrClient, addrServ2;
    socklen_t lenAddrClient;

    if ((sockServ1 = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
    perror("Erreur socket");
    exit(1);
    }
    if ((sockServ2 = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
    perror("Erreur socket");
    exit(1);
    }

    bzero(&monAddr, sizeof(monAddr));
    monAddr.sin_family = AF_INET;
    monAddr.sin_port = htons(ports.port1);
    monAddr.sin_addr.s_addr = INADDR_ANY;
    bzero(&addrServ2, sizeof(addrServ2));
```

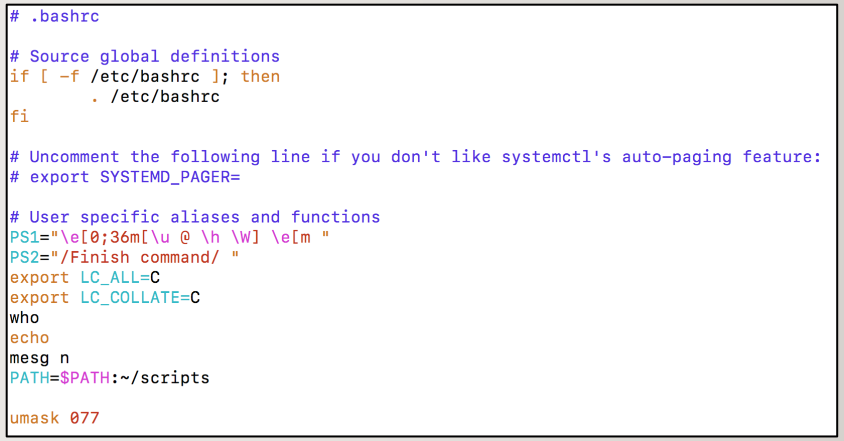
CREATING TEXT FILES

Text Editors

Networking and Tech Support students use a text editor to **edit configuration files**.

Throughout their program, students will become familiar with the process of **installing**, **configuring**, and **running** network services on their Linux servers.

Text editors are an important tools to help setup but also "**tweak**" or make **periodic changes in networking services configuration**.



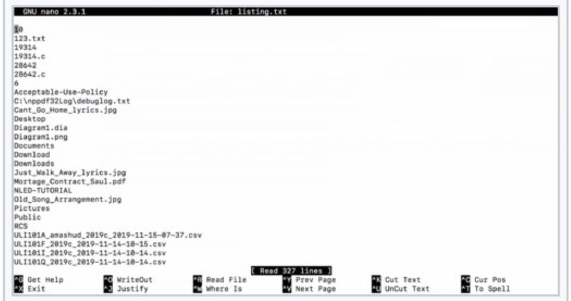
```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions
PS1="\e[0;36m[\u @ \h \W] \e[m "
PS2="/Finish command/ "
export LC_ALL=C
export LC_COLLATE=C
who
echo
mesg n
PATH=$PATH:~/scripts

umask 077
```

CREATING TEXT FILES

Text Editors

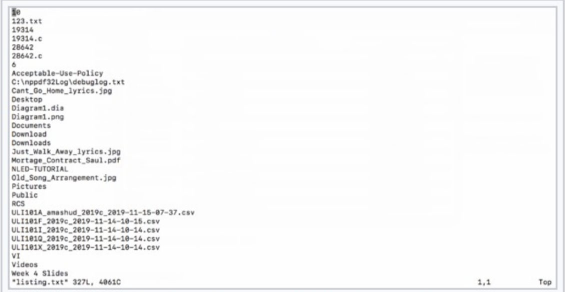
Regardless of the IT stream that they are in, it is useful for students to **expose themselves to different text editors and then use one that they feel most comfortable working with**.

The two most readily-available command line text editors in Linux are **`nano`** and **`vi`**.



```
  GNU nano 2.3.1                   File: listing.txt

10
123.txt
19314
19314.c
28642
28642.c
6
Acceptable-Use-Policy
C:\nppdf32Log\debuglog.txt
Cant_Go_Home_lyrics.jpg
Desktop
Diagram1.dia
Diagram1.png
Documents
Download
Downloads
Just_Walk_Away_lyrics.jpg
Mortage_Contract_Saul.pdf
NLED-TUTORIAL
Old_Song_Arrangement.jpg
Pictures
Public
RCS
ULI101A_amashud_2019c_2019-11-15-07-37.csv
ULI101F_2019c_2019-11-14-10-15.csv
ULI101I_2019c_2019-11-14-10-14.csv
ULI101Q_2019c_2019-11-14-10-14.csv
                                    [ Read 327 lines ]
^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

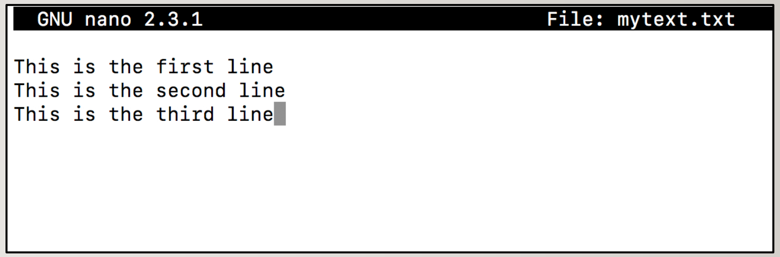
The **Nano Text Editor** is Easy to Operate for Novice Users.



```
10
123.txt
19314
19314.c
28642
28642.c
6
Acceptable-Use-Policy
C:\nppdf32Log\debuglog.txt
Cant_Go_Home_lyrics.jpg
Desktop
Diagram1.dia
Diagram1.png
Documents
Download
Downloads
Just_Walk_Away_lyrics.jpg
Mortage_Contract_Saul.pdf
NLED-TUTORIAL
Old_Song_Arrangement.jpg
Pictures
Public
RCS
ULI101A_amashud_2019c_2019-11-15-07-37.csv
ULI101F_2019c_2019-11-14-10-15.csv
ULI101I_2019c_2019-11-14-10-14.csv
ULI101Q_2019c_2019-11-14-10-14.csv
ULI101X_2019c_2019-11-14-10-14.csv
VI
Videos
Week 4 Slides
"listing.txt" 327L, 4061C                                          1,1          Top
```

The Vi Text Editor, although taking longer to master, has outstanding features and allow the user to be more productive.

CREATING TEXT FILES



```
GNU nano 2.3.1                                              File: mytext.txt

This is the first line
This is the second line
This is the third line
```

Nano Text Editor

The **nano** text editor is considered to be an easy-to-use text editor. When using the nano text editor, you are placed in **INPUT** mode, to enter text immediately.

Nano editing **commands** typically consist of the **^** symbol which represents the **`<ctrl>`** key followed by a character.

NOTE: There is no **undo** command in Nano!

The table on the right list a few Nano commands and their purpose. Refer to **week 2 notes** for a **nano reference sheet.**

NOTE: In the Nano reference sheet, the letter **`M`** represents the **`<esc>`** key

Key Combination	Purpose
`<ctrl><space>` , **`<esc><space>`**	Move forward / backward one word
`<ctrl>a` , **`<ctrl>e`**	Move to beginning / end of line
`<ctrl>k`	Cut line
`<esc>6`	Copy Line
`<ctrl>u`	Paste Cut / Copied Text
`<ctrl>g`	Display help screen
`<ctrl>x`	Save and exit editing session

MANAGING DIRECTORIES

Instructor Demonstration

Your instructor will demonstrate how to create and edit a text file using the nano text editor.

CREATING TEXT FILES

`vi` Text Editor

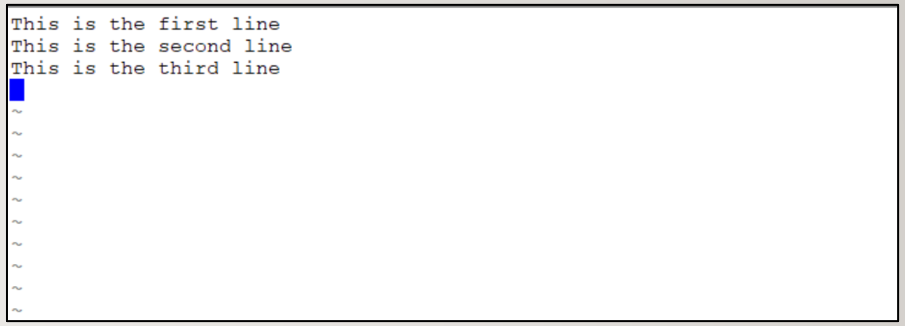
The **vi** (**vim**) text editor (although taking longer to learn) has outstanding features to increase coding productivity.

The major different between nano and vi is that **vi starts in COMMAND LINE mode**. You need to issue letter commands to perform text editing or press colon ":" to enter last line mode to issue more complex commands.

To make it easier to learn how to use this text editor, an **online tutorial** was created (two decades ago) to provide you "hands-on" experience in command editing techniques.

To run this tutorial, issue the following command in Matrix:
/home/jason.carman/vi-tutorial

You can refer to your **week 2 notes** for a **vi command reference sheet**.



```
This is the first line
This is the second line
This is the third line

~
~
~
~
~
~
~
~
~
~
```

Key Combination	Purpose
`i`	Enter **INSERT** mode
`<esc>`	Return to COMMAND mode
`B , W`	Move forward / backward one word
`0 , $`	Move to beginning / end of line
`dd`	Cut line
`yy`	Copy Line
`p , P`	Paste below / above line
`:help`	Display help screen
`:x`	Save and exit editing session

MANAGING DIRECTORIES

Instructor Demonstration

Your instructor will demonstrate how to create and edit a text file using the **vi** text editor.

MANAGING TEXT FILES

Purpose

It is **essential** for students in this course not only to create text files but also to learn how to **manage** text files.

Students need to learn how to **create** empty files, **copy** files for backup purposes, **move** or **rename** incorrectly spelled filenames, **edit** files as well as **view** text file contents without the danger of editing or corrupting those files.

Students also need to learn how to **remove** files, check for **differences** between a couple of files as well as **obtain information** regarding the status of a file and information regarding the file's content.

MANAGING TEXT FILES

Text File Management Commands

Here are common text file management commands:

Linux Command	Purpose
`touch`	Create empty file(s) / Updates Existing File's Date/Time Stamp
`cat`	Display text file's contents without editing (small files)
`more , less`	Display / Navigate within large text files without editing
`head , tail`	View lines at top/bottom of file
`grep`	Display lines in file that match a pattern
`cp`	Copy text file(s)
`mv`	Move / Rename text files
`rm`	Remove text file(s)
`diff`	Displays differences between 2 files

MANAGING TEXT FILES

Text File Management Commands

Here are some **additional** text file management commands:

Linux Command	Purpose
`sort`	Display contents of file in sorted order
`uniq`	Display identical adjacent lines only once
`file`	Gives info about the contents of the file (e.g. file with no extension)

`find`	To find files matching specified characteristics:
	
	`find . -name "file*”`
	lists pathname of any filenames beginning with "file", from the current directory and any subdirectories
	`find . -size +50k`
	lists pathname of any files larger than 50 kb, from the current directory and any subdirectories
	`find . -mmin -5`
	lists files modified less than 5 minutes ago

MANAGING DIRECTORIES

Managing Manipulating Text Files

Your instructor will demonstrate how to **manage** / **manipulate** text files:

- Create empty files
- View small and large text files
- Sort files
- Display matched pattern file content
- Remove duplicate lines
- Compare files for differences
- Obtain file information / List file pathnames

HOMEWORK

Getting Practice

Perform the online tutorial **Tutorial2: Unix / Linux File Management**

(Due: Friday Week 3 @ midnight for a 2% grade):

- INVESTIGATION 2: MANAGING TEXT FILES
- LINUX PRACTICE QUESTIONS (Questions 9 – 16)