**System Programming**

#3

**Comparison of low-level vs. high-level file access**

**1분반**

**장경식 교수님**

**2012136020 김동우**

**2016.10.07**

**코드에 관한 자세한 내용은 소스파일에 있습니다.**

==========================================

복사할 파일을 따로 리눅스에 관한 내용들을 정리한 텍스트 파일을 사용하였다.

**LowCopy & HighCopy**

**Low-level**

시스템 콜(system call)을 통한 접근

File Descriptor(파일 접근에 사용되는 정수값)을 활용

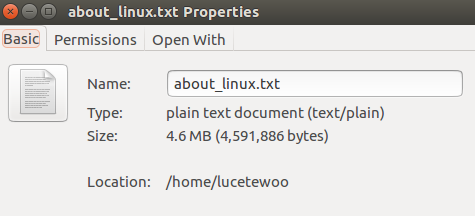
**High-level**

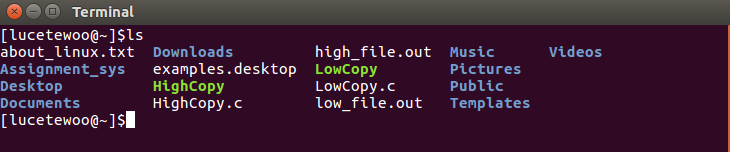
표준 입출력 라이브러리를 사용한 파일 함수

File Descriptor 대신 File stream을 사용

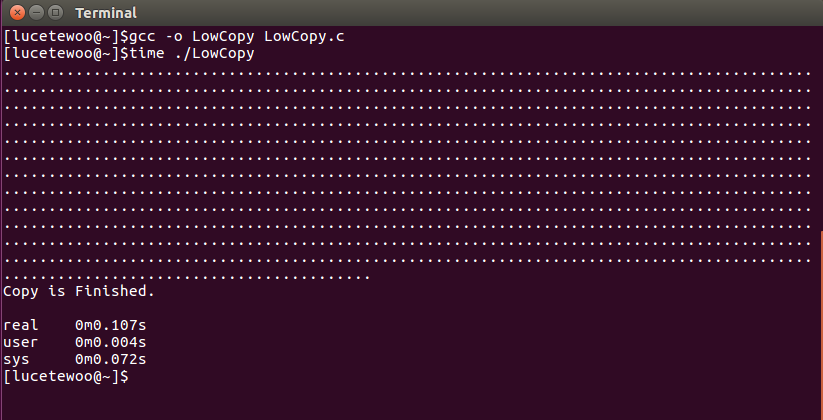
File stream : 데이터 구조체 FILE에 대한 포인터 (FILE\*)을 의미한다.

C언어에서 사용하던 fopen, fclose, fread, fwrite등의 고수준 파일 함수를 사용할 수 있다.

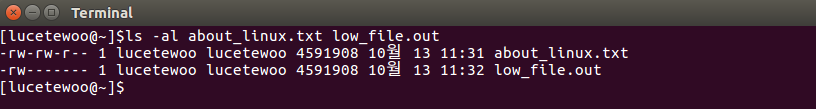




**LowCopy**

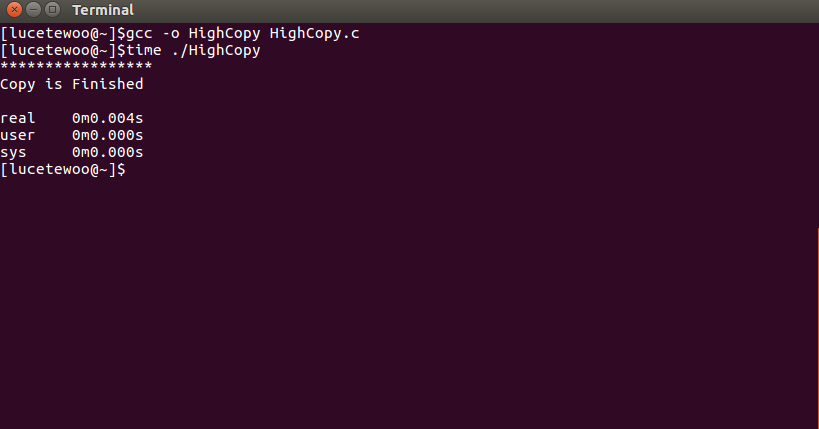


LowCopy파일을 time명령어를 활용하였다.

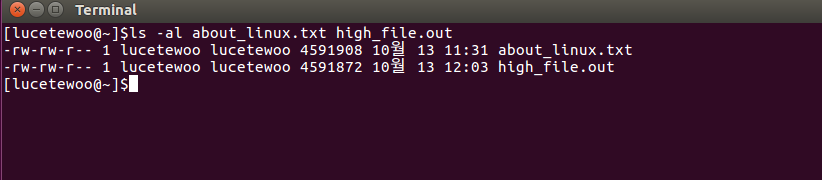


코드를 짤 때 복사 대상 파일로 설정했던 about\_linux.txt을 복사하여 low\_file.out이라는 파일이 생성되었다.

**HighCopy**



LowCopy와 마찬가지로 time 명령어를 사용하여 실행하였다. Asterisk(\*)을 출력하도록 하였다.



마찬가지로 about\_linux.txt파일을 복사하여 high\_file.out 라는 파일을 복사하였다. rw설정을 사용하여, 접근권한이 동일하게 생성이 되었다.

LowCopy는 0.107s, HighCopy는 0.04s로 엄청난 실행 속도 차이를 보였다. LowCopy에서는 배열을 통하여 버퍼를 생성하였고, HighCopy에서는 stdio 라이브러리가 내부 버퍼를 FILE 구조체에 유지하고 버퍼가 가득 찬 경우에만 저수준 시스템 호출을 수행하기 때문이다.

**느낀점**

: 리눅스를 활용하여 프로그래밍을 계속 진행하다보니, 이제 vi편집기, 리눅스 명령어에 대해서는 많은 부분 익숙해 졌다는 생각이 들었다. 코드의 모든 부분들을 스스로 짜기에는 아직 어려움이 많지만, 책, 프린트물 그리고 다른사람들의 코드와 방법을 참고하여 스스로 virtual machine을 활용해 코드도 짜보고 실행해보니 많은 부분에 대해 알게 된 것 같다.