

Forum: [Hack Forums](#)

Thread: [C++ with large files.](#)

severos - June 21, 2021, 9:19 pm

I have to deal with files of very large size, 100GB+, and the main problem is in seeking around those files since the normal fseek takes 32bit offset which doesn't cover the whole file.

That's when I found the functions: fopen64, fseeko64, ftello64 that supposedly allow you to use 64bit offset...

But for the last 2 days I can't get them to work, they do work as in, file opens, seek, read/write as long as the offset is within 32bit range, but soon as I try to seek beyond that fseeko64 fails and the offset is always at 0.

I did:

```
#define _FILE_OFFSET_BITS 64
#define __USE_FILE_OFFSET64 1
#define _GNU_SOURCE 1
```

But it still doesn't work, anything beyond the 32bits offset places the offset at 0.

Any idea what could be the problem?

HFDone - June 22, 2021, 2:43 am

you can
reading chunks
using file map api
using istream seekg
google

severos - June 22, 2021, 3:51 am

(Jun 22, 2021 02:43 AM)HFDone Wrote: [\[CITE\]](#) you can
reading chunks
using file map api
using istream seekg
google

Reading chunks: worst solution ever, it'll require reading data that I won't use, performance will be super slow. Never suggest that.

File map: might be useful in other scenarios, not in my case, I'm extending the file as I read/write to it, mmf require constant file size.

istream seekg: according to my original post: "main problem is in seeking around those files since the normal fseek takes 32bit offset which doesn't cover the whole file", so seekg is actually not possible, maximum you can reach with that is 4GB.

Google: that's how I got to what I mentioned in my original post "fopen64, fseeko64, ftello64", however my google skills has failed in debugging why they keep failing. Which is why I posted here.

Sajuuk - June 22, 2021, 11:05 am

<https://www.cplusplus.com/reference/string/string/getline/>

The margin of efficiency difference in parsing massive chunks of data is negligible. Some methods

may hold out better in specific cases, but in terms of massive data parsing of some generic data (generic as in specified as "100+ gB of Data") speed will be matched based on system specs unless some specific data type is specified.

severos - June 22, 2021, 3:50 pm

(Jun 22, 2021 11:05 AM)Sajuuk Wrote: [\[CITE\]](#)
<https://www.cplusplus.com/reference/string/string/getline/>

The margin of efficiency difference in parsing massive chunks of data is negligible. Some methods may hold out better in specific cases, but in terms of massive data parsing of some generic data (generic as in specified as "100+ gB of Data") speed will be matched based on system specs unless some specific data type is specified.

Working with binary files, and I don't want to read the whole file. Again, my problem is in seeking after the 4GB barrier.

HFDone - June 23, 2021, 12:52 pm

did you check SetFilePointerEx

<https://docs.microsoft.com/he-il/windows/win32/api/fileapi/nf-fileapi-setfilepointerex>
you can use it with CreateFile

severos - June 23, 2021, 3:51 pm

(Jun 23, 2021 12:52 PM)HFDone Wrote: [\[CITE\]](#) did you check SetFilePointerEx
<https://docs.microsoft.com/he-il/windows/win32/api/fileapi/nf-fileapi-setfilepointerex>
you can use it with CreateFile

My bad, forgot to mention that I'm coding on linux not windows. But thanks for the contribution.

int main() - June 24, 2021, 5:17 pm

ulimit -n 9999999 i think would help

tulamba - June 26, 2021, 1:37 pm

@serveros

Have you looked at multithreading or multiprocessing options to split the task into smaller subtasks?

Can you share the code?

severos - June 26, 2021, 7:48 pm

(Jun 24, 2021 05:17 PM)int main() Wrote: [\[CITE\]](#) ulimit -n 9999999 i think would helpI can't even think of how that would even be remotely related to my question?

(Jun 26, 2021 01:37 PM)tulamba Wrote: [\[CITE\]](#) @serveros

Have you looked at multithreading or multiprocessing options to split the task into smaller subtasks?
Unrelated solution, I still need to seek through the file regardless of how I process it.

tulamba - June 26, 2021, 8:01 pm

<https://stackoverflow.com/questions/34751873/how-to-read-huge-file-in-c>