

【C++ IO机制】stream_buf 解析

目录

- stream_buf 解析
- 参考资料

正文

回到顶部

stream_buf 解析

1. stream_buf原型:

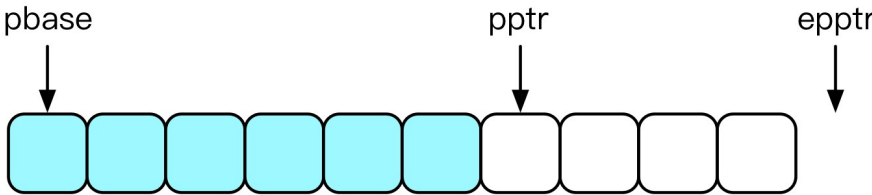
```
template <class CharT, class Traits = std::char_traits<CharT>>
class basic_streambuf;
```

【注意】

```
typedef basic_streambuf<char> streambuf;
```

用于输出的 streambuf

streambuf 使用三个指针来管理相应的输出缓冲区(缓冲区需要自行设置), 分别由接口 pbase, pptr 和 epptr 返回。其中 pbase 是缓冲区的基指针, 指向缓冲区的第一个字节, epptr 是缓冲区的尾指针, 指向其最后一个字节的下一个字节(类似于 iter.end() 的作用), 而 pptr 指向缓冲区当前可用的位置, 也就是pptr 之前都已经被数据所填充, 如下图:

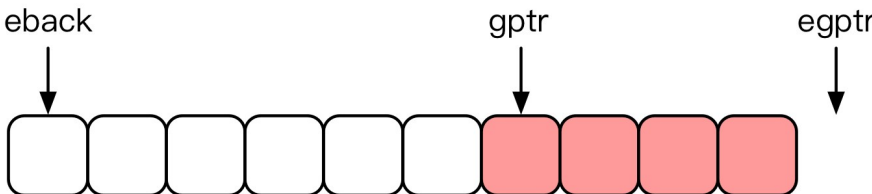


输出缓冲区

streambuf 定义的输出相关的函数主要有 sputc 和 sputn, 前者输出一个字符到缓冲区, 并且将指针 pptr 向后移动一个字符, 后者调用函数 xsputn 连续输出多个字符, xsputn 默认的实现就是多次调用 sputc, 由于缓冲区有限, 当 pptr 指针向后移动满足 pptr() == epptr 时, 说明缓冲区满了, 这时将会调用函数 overflow 将数据写入到外部设备并清空缓冲区;清空缓冲区的方式则是调用 pbump 函数将指针 pptr 重置。我们可以通过如下的类来实现自定义的输出 buffer:

用于输入的 streambuf

同管理输出缓冲区一样, streambuf 也使用三个指针, eback(), gptr() 以及 egptr() 分别指示输入缓冲区的开始字节, 当前可用字节以及缓冲区尾的下一字节, 如下图所示:



输入缓冲区

streambuf 类同样定义了如下几个函数来支持对于输入缓冲区的读取和管理:

- sgetc: 从输入缓冲区中读取一个字符;
- sbumpe: 从输入缓冲区中读取一个字符, 并将 gptr() 指针向后移动一个位置;
- sgetn: 从输入缓冲区中读取 n 个字符;
- sungetc: 将缓冲区的 gptr() 指针向前移动一个位置;
- sputback: 将一个读取到的字符重新放回输入缓冲区中;

与输出缓冲区不同的是, 输入缓冲区需要额外提供 putback 操作, 也就是将字符放回输入缓冲区内。

1. 示例

```
#include <iostream>
#include <fstream>

int main()
{
    std::ifstream ifs("test.txt");
    if (ifs.good())
    {
        std::streambuf *pbuf = ifs.rdbuf();
        char c;
        ifs >> c;
        std::streamsize size = pbuf->in_avail();
        std::cout << "first character in file: " << c << '\n';
        std::cout << size << " characters in buffer after it\n";
    }
    ifs.close();
    return 0;
}
```

说明: test.file 文件内容如下:

```
this is test file!
```

输出:

随笔 - 535 文章 - 0 评论 - 4 阅读 - 12万

公告

昵称: 苏格拉底的落泪
园龄: 5年2个月
粉丝: 9
关注: 31
+加关注

< 2023年3月 >						
日	一	二	三	四	五	六
26	27	28	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

搜索



常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签

最新随笔

1. ext2文件系统
2. 原子操作_atomic 系列函数
3. coreddumpctl
4. nfs-ganesha nfs4.1协议数据恢复机制
5. The user-space RCU API
6. 设置时间同步
7. nfs-ganesha NFS4.1 rados_cluster数据恢...
8. 如何永久配置cephfs mds热备
9. cephfs client evict子命令使用
10. C语言类型转换常用函数

我的标签

力扣(98)
C++基础(29)
linux系统编程(26)
git(25)
linux命令(20)
ceph运维(15)
osd(14)
C++继承(14)
ceph集群(13)
cmake(10)
更多

随笔分类

- 【01】C 基础(6)
- 【02】C++ 基础(67)
- 【03】C++ 多线程(11)
- 【04】C++ 模板(11)
- 【05】C++ boost库(5)
- 【06】linux网络编程(6)
- 【07】linux系统编程(24)
- 【08】linux命令(25)
- 【09】shell脚本(1)
- 【10】力扣(100)
- 【11】make(14)
- 【12】gdb(1)
- 【13】git(28)
- 【14】docker(9)
- 【15】makefile(8)
- 更多

随笔档案

- 2023年3月(3)
- 2023年2月(2)
- 2023年1月(10)
- 2022年12月(3)
- 2022年11月(3)
- 2022年10月(4)
- 2022年9月(1)
- 2022年8月(2)
- 2022年7月(16)
- 2022年6月(13)
- 2022年5月(85)
- 2022年4月(32)
- 2022年3月(9)
- 2022年2月(6)
- 2022年1月(6)
- 更多

阅读排行榜

1. 字节对齐(内存对齐)(7966)
2. 宏offsetof(6494)
3. 【力扣 002】114. 二叉树展开为链表(4865)
4. compare_exchange_strong 函数解析(3315)
5. 【C++ Primer | 15】虚继承(2399)

```
first character in file: t
18 characters in buffer after it
```

```
1 #include <iostream>
2 #include <sstream>
3
4 int main()
5 {
6     std::stringstream stream("Hello, world");
7     std::cout << "sgetc() returned '" << (char)stream.rdbuf()->sgetc() << "'\n";
8     std::cout << "peek() returned '" << (char)stream.peek() << "'\n";
9     std::cout << "get() returned '" << (char)stream.get() << "'\n";
10 }
```

输出:

```
1 sgetc() returned 'H'
2 peek() returned 'H'
3 get() returned 'H'
```

说明:

sgetc()	返回当前位置的字符。
sputc()	返回当前位置的字符, 并将当前位置前进到下一个字符。
snextc()	将当前位置前进到下一个字符并返回下一个字符。

回到顶部

参考资料

- std::streambuf从示例到应用
- stream_buff 解析【cppreference.com】
- C/C++编程:流缓冲类std::basic_streambuf

分类: 【02】C++ 基础

标签: C++ IO

好文要顶 关注我 收藏该文

苏格拉底的薄泪
粉丝 - 9 关注 - 31

+加关注

0

推荐

0

反对

- « 上一篇: 【C++ IO机制】文件输入输出 ①
» 下一篇: 【C++ IO机制】标准输入输出 ①

posted @ 2021-12-25 13:59 苏格拉底的薄泪 阅读(482) 评论(0) 编辑 收藏 举报

最新评论 刷新页面 返回顶部

登录后才能查看或发表评论, 立即 登录 或者 注册 博客园首页

【推荐】阿里云2核2G云服务器低至99元年, 百款云产品优惠享不停

编辑推荐:

- NET Task 揭秘: async 与 AsyncMethodBuilder
- 记一次 .NET 算汽车零件采集系统 卡死分析
- 关于如何编写好金融科技客户端 SDK 的思考
- ASP.NET Core Web API 接口规范
- 生产环境 Java 应用服务内存泄漏分析与解决

阅读排行:

- 为什么 C# 可能是最好的第一编程语言
- 我的十年编程路 2022年篇
- 震惊, 一行MD5居然让小伙伴们回不了家!!!
- .NET 8 预览版 2 亮点是Blazor
- 我的十年编程路 尾声

评论排行榜

1. 【ceph运维】删除osd和host(2)
2. 【ceph集群】docker 安装 ceph 集群(1)
3. 【C++ Primer 第15章】定义派生类拷贝...

推荐排行榜

1. 【osd】OSD的内部队列(1)
2. install(1)
3. 【linux命令】查询命令安装路径(1)
4. epoll(二)(1)
5. 字节对齐(内存对齐)(1)

最新评论

1. Re:【ceph集群】docker 安装 ceph 集群
都用了三台虚拟机, 还用docker干啥? 不会ceph部署吗? --歪正中
2. Re:【ceph | 运维】删除osd和host
@xu_jd 查看一下 /usr/lib/systemd/system 目录下是否存在 ceph-osd@.service 文件, 这个文件在ceph源码的systemd目录下, 我的操作系统为c... --苏格拉底的薄泪
3. Re:【ceph | 运维】删除osd和host
为啥我的15.2没有这样的命令呢? [root@ceph1 ceph]# ceph osd tree ID CLASS WEIGHT TYPE NAME STATUS REWEIGHT PRI-AF... --xu_jd
4. Re:【C++ Primer 第15章】定义派生类拷...
天才呀, 看了此文, 恍然大悟 --追梦的路上

