

C++程序设计基础 第十二章

前言

第十二章 I/O流

12.1 I/O流类库

12.1.1 ios类库

12.1.2 streambuf类库

12.2 标准I/O流

12.2.1 预定义流对象

12.2.2 标准输出流

1. put()函数

2. write()函数

12.2.3 标准输入流

1. get()函数

前言

本文档由 @ItsJiale 创作，作者博客：<https://jiale.domcer.com/>，作者依据数学与大数据学院 2024 级大数据教学班的授课重点倾向编写而成。所有内容均为手动逐字录入，其中加上了不少自己的理解与思考，耗费近一周时间精心完成。

此文档旨在助力复习 C++ 程序设计基础，为后续学习数据结构筑牢根基。信计专业的同学，也可参考本文档规划复习内容。需注意，若个人学习过程中存在不同侧重点或对重难点的理解有差异，应以教材内容为准。倘若文档内容存在任何不妥之处，恳请各位读者批评指正。

By: ItsJiale

2025.5.5

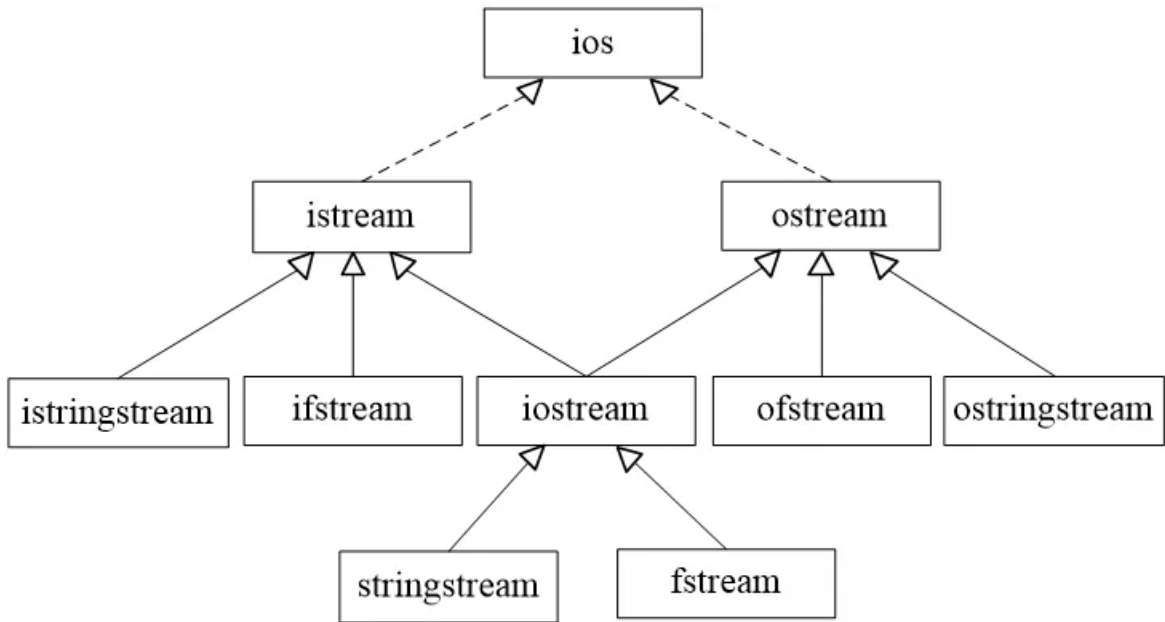
第十二章 I/O流

12.1 I/O流类库

I/O流库是C++标准库的重要组成部分，它主要包括 `ios` 类库和 `streambuf` 类库。其中，`ios` 类库提供流的高级I/O操作，`streambuf` 类库主要负责缓冲区的处理。

12.1.1 ios类库

`ios` 类库以 `ios` 类为根基类，`ios` 类是一个抽象类（包含纯虚函数的类叫做抽象类），提供了输入/输出所需的公共接口，如设置数据流格式、错误状态恢复、设置文件的输入/输出模式等。`ios` 类库层次结构如下



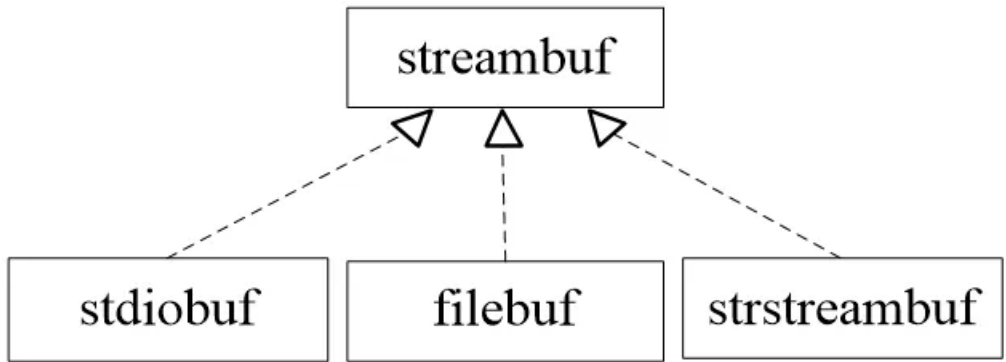
抽象基类 `ios` 类派生了2个类：`istream` 类和 `ostream` 类。`istream` 类和 `ostream` 类是输入输出流类，定义了输入流和输出流的基本特性。`istream` 类和 `ostream` 类又派生了其他类。

`ios` 类库中各个类的含义与功能如下所示：

- `ifstream`类：文件输入流类，支持文件的读取操作。
- `istringstream`类：字符串输入流类，支持字符串的输入操作。
- `ofstream`类：文件输出流类，支持文件的写入操作。
- `ostringstream`类：字符串输出流类，支持字符串的输出操作。
- `fstream`类：文件输入/输出流类，支持文件的读取和写入操作。
- `stringstream`类：字符串输入/输出流为，支持字符串输入和输出操作。

12.1.2 streambuf类库

`streambuf` 类库以 `streambuf` 为根基类，`streambuf` 类是一个抽象类，提供了缓冲区操作接口，如设置缓冲区、从缓冲区提取字节、向缓冲区插入字节等。`streambuf` 类库层次结构如下图。



12.2 标准I/O流

12.2.1 预定义流对象

C++提供了4个预定义流对象：`cin`、`cout`、`cerror` 和 `clog`。`cin` 是 `istream` 类的对象，用于处理标准输入（键盘输入）。`cout`、`cerror` 和 `clog` 是 `ostream` 类的对象，`cout` 用于处理标准输出（屏幕输出）；`cerr` 和 `clog` 用于处理标准错误信息。

对象名	所属类	对应设备	含义
cin	istream	键盘	标准输入，有缓冲
cout	ostream	屏幕	标准输出，有缓冲
cerr	ostream	屏幕	标准错误输出，无缓冲
clog	ostream	屏幕	标准错误输出，有缓冲

12.2.2 标准输出流

`ostream` 预定义了三个输出流对象 `cout`、`cerr` 和 `clog`，而且还重载了运算符“<<”，输出流对象与“<<”运算符结合使用，可以输出各种类型的数据。此外，`ostream` 类还提供了成员函数输出数据，比较常用的两个函数为 `put()` 函数和 `write()` 函数。

1. put()函数

`put()` 函数用于输出单个字符，其声明如下所示：

```
ostream& put(char ch);
```

在上述声明中，参数 `ch` 表示要输出的字符，函数返回值为 `ostream` 类对象引用。`put()` 函数的作用是将字符插入到输出流对象，通过输出流对象将字符输出到指定位置。

由于 `put()` 函数返回的是输出流对象，因此 `put()` 函数与输出运算符 “<<” 一样，可以连续调用。

```
cout.put('a'); //输出字符a
```

```
cout.put('\n'); //输出换行符
```

```
cout.put('d').put('h'); //连续调用put()函数，输出‘ d’、‘h’
```

2. write()函数

`write()` 函数用于输出一个字符串，其声明如下所示：

```
ostream& write(const char* str, streamsize count);
```

在上述声明中，第一个参数 `str` 表示字符串，第二个参数 `count` 表示输出的字符个数，需要注意的是，`streamsize` 是 `long long` 类型的重定义。`write()` 函数返回值为 `ostream` 类对象引用，其作用是将指定个数的字符串插入到输出流对象，通过输出流对象将字符串输出到指定位置。`write()` 函数也可以连续调用。

```
cout.write("I love China",6);
```

```
cout.write("I love China",6).write("I love China",5);
```

在上述代码中，第一行代码调用 `write()` 函数输出字符串 “I love China” 的前 6 个字符；第二行代码连续调用 `write()` 函数输出字符串 “I love China” 的前 6 个和前 5 个字符。

还是那句话，会用就行

12.2.3 标准输入流

`istream` 类预定义了输入流对象 `cin`，并且重载了运算符 “>>”，输入流对象与 “>>” 运算符结合使用，可以输入各种类型的数据。此外，`istream` 类还提供了成员函数输入数据，如 `get()` 函数、`getline()` 函数、`read()` 函数等。

1. get()函数

`get()` 函数用于从输入流中获取单个字符，`istream` 类提供了 `get()` 函数的三种重载形式。

(1) 读取单个字符

从流中读取单个字符的 `get()` 函数有两种重载形式，具体如下所示：

```
int get(); //第一种形式
```

```
istream& get(char& ch); //第二种形式
```

上述函数声明中，第一种形式没有参数，返回值为 `int` 类型，函数的作用是从输入流读取一个字符，返回该字符的 `ASCII` 码值。第二种形式有一个 `char` 类型的引用作为参数，返回值为 `istream` 类对象引用。函数的作用是从输入流读取一个字符存储到 `ch` 中。