C++ 类成员访问运算符 -> 重载 C++ 异常处理

C++ 文件和流

到目前为止，我们已经使用了 **iostream** 标准库，它提供了 **cin** 和 **cout** 方法分别用于从标准输入读取流和向标准输出写入流。

本教程介绍如何从文件读取流和向文件写入流。这就需要用到 C++ 中另一个标准库 **fstream**，它定义了三个新的数据类型：

数据类型	描述
ofstream	该数据类型表示输出文件流，用于创建文件并向文件写入信息。
ifstream	该数据类型表示输入文件流，用于从文件读取信息。
fstream	该数据类型通常表示文件流，且同时具有 ofstream 和 ifstream 两种功能，这意味着它可以创建文件，向文件写入信息，从文件读取信息。

要在 C++ 中进行文件处理，必须在 C++ 源代码文件中包含头文件 <iostream> 和 <fstream>。

打开文件

在从文件读取信息或者向文件写入信息之前，必须先打开文件。**ofstream** 和 **fstream** 对象都可以用来打开文件进行写操作，如果只需要打开文件进行读操作，则使用 **ifstream** 对象。

下面是 open() 函数的标准语法，open() 函数是 fstream、ifstream 和 ofstream 对象的一个成员。

```
void open(const char *filename, ios::openmode mode);
```

在这里，**open()** 成员函数的第一参数指定要打开的文件的名称和位置，第二个参数定义文件被打开的模式。

模式标志	描述
ios::app	追加模式。所有写入都追加到文件末尾。
ios::ate	文件打开后定位到文件末尾。
ios::in	打开文件用于读取。
ios::out	打开文件用于写入。
ios::trunc	如果该文件已经存在，其内容将在打开文件之前被截断，即把文件长度设为 0。

您可以把以上两种或两种以上的模式结合使用。例如，如果您想要以写入模式打开文件，并希望截断文件，以防文件已存在，那么您可以使用下面的语法：

```
ofstream outfile;
outfile.open("file.dat", ios::out | ios::trunc );
```

类似地，您如果想要打开一个文件用于读写，可以使用下面的语法：

```
ifstream  afile;
afile.open("file.dat", ios::out | ios::in );
```

关闭文件

当 C++ 程序终止时，它会自动关闭刷新所有流，释放所有分配的内存，并关闭所有打开的文件。但程序员应该养成一个好习惯，在程序终止前关闭所有打开的文件。

下面是 close() 函数的标准语法，close() 函数是 fstream、ifstream 和 ofstream 对象的一个成员。

```
void close();
```

写入文件

在 C++ 编程中，我们使用流插入运算符（ << ）向文件写入信息，就像使用该运算符输出信息到屏幕上一样。唯一不同的是，在这里您使用的是 **ofstream** 或 **fstream** 对象，而不是 **cout** 对象。

读取文件

在 C++ 编程中，我们使用流提取运算符（ >> ）从文件读取信息，就像使用该运算符从键盘输入信息一样。唯一不同的是，在这里您使用的是 **ifstream** 或 **fstream** 对象，而不是 **cin** 对象。

读取 & 写入实例

下面的 C++ 程序以读写模式打开一个文件。在向文件 afile.dat 写入用户输入的信息之后，程序从文件读取信息，并将其输出到屏幕上：

实例

```
#include <fstream>
#include <iostream>
using namespace std;
int main ()
{
char data[100];
// 以写模式打开文件
ofstream outfile;
outfile.open("afile.dat");
cout << "Writing to the file" << endl;
cout << "Enter your name: ";
cin.getline(data, 100);
// 向文件写入用户输入的数据
outfile << data << endl;
cout << "Enter your age: ";
cin >> data;
cin.ignore();
// 再次向文件写入用户输入的数据
outfile << data << endl;
```

```
// 关闭打开的文件
outfile.close();
// 以读模式打开文件
ifstream infile;
infile.open("afile.dat");
cout << "Reading from the file" << endl;
infile >> data;
// 在屏幕上写入数据
cout << data << endl;
// 再次从文件读取数据，并显示它
infile >> data;
cout << data << endl;
// 关闭打开的文件
infile.close();
return 0;
}
```

当上面的代码被编译和执行时，它会产生下列输入和输出：

```
$./a.out
Writing to the file
Enter your name: Zara
Enter your age: 9
Reading from the file
Zara
9
```

上面的实例中使用了 cin 对象的附加函数，比如 getline()函数从外部读取一行，ignore() 函数会忽略掉之前读语句留下的多余字符。

文件位置指针

istream 和 **ostream** 都提供了用于重新定位文件位置指针的成员函数。这些成员函数包括关于 istream 的 **seekg**（"seek get"）和关于 ostream 的 **seekp**（"seek put"）。

seekg 和 seekp 的参数通常是一个长整型。第二个参数可以用于指定查找方向。查找方向可以是 **ios::beg**（默认的，从流的开头开始定位），也可以是 **ios::cur**（从流的当前位置开始定位），也可以是 **ios::end**（从流的末尾开始定位）。

文件位置指针是一个整数值，指定了从文件的起始位置到指针所在位置的字节数。下面是关于定位 "get" 文件位置指针的实例：

```
// 定位到 fileObject 的第 n 个字节（假设是 ios::beg）
fileObject.seekg( n );
// 把文件的读指针从 fileObject 当前位置向后移 n 个字节
fileObject.seekg( n, ios::cur );
// 把文件的读指针从 fileObject 末尾往回移 n 个字节
fileObject.seekg( n, ios::end );
// 定位到 fileObject 的末尾
fileObject.seekg( 0, ios::end );
```

C++ 类成员访问运算符 -> 重载 C++ 异常处理

5 篇笔记 **写笔记**