* 对自己的爱好保持热情，不要太功利

@[toc]

# 文件操作

**文件类型分为两种**：

1. **文本文件** : 文件以文本的**ASCII码**形式存储在计算机中
2. **二进制文件** : 文件以文本的**二进制**形式存储在计算机中，用户一般不能直接读懂它们

* 程序运行时产生的数据都属于临时数据，程序一旦运行结束都会被释放

通过**文件可以将数据持久化**

C++中对文件操作需要包含头文件 &lt; fstream &gt;

**操作文件的三大类**:：

1. ofstream：写操作
2. ifstream： 读操作
3. fstream ： 读写操作

## 文本文件

### 写文件

写文件步骤如下：

1. 包含头文件 #include <fstream>
2. 创建流对象 ofstream ofs;
3. 打开文件 ofs.open("文件路径",打开方式);
4. 写数据 ofs << "写入的数据";
5. 关闭文件 ofs.close();

文件打开方式：

|  |  |
| --- | --- |
| 打开方式 | 解释 |
| ios::in | 为读文件而打开文件 |
| ios::out | 为写文件而打开文件 |
| ios::ate | 初始位置：文件尾 |
| ios::app | 追加方式写文件 |
| ios::trunc | 如果文件存在先删除，再创建 |
| ios::binary | 二进制方式 |

**注意：** 文件打开方式可以配合使用，利用|操作符

**例如** ： 用二进制方式写文件 ios::binary | ios:: out  
**示例：**

#include <fstream>  
  
void test01()  
{  
 ofstream ofs;  
 ofs.open("test.txt", ios::out);  
  
 ofs << "姓名：张三" << endl;  
 ofs << "性别：男" << endl;  
 ofs << "年龄：18" << endl;  
  
 ofs.close();  
}  
  
int main() {  
  
 test01();  
  
 system("pause");  
  
 return 0;  
}

总结：

* 文件操作必须包含头文件 fstream
* 读文件可以利用 ofstream ，或者fstream类
* 打开文件时候需要指定操作文件的路径，以及打开方式
* 利用<<可以向文件中写数据
* 操作完毕，要关闭文件

### 读文件

读文件与写文件步骤相似，但是读取方式相对于比较多

读文件步骤如下：

1. 包含头文件 #include <fstream>
2. 创建流对象 ifstream ifs;
3. 打开文件并判断文件是否打开成功 ifs.open("文件路径",打开方式);
4. 读数据 : 四种方式读取
5. 关闭文件 ifs.close();

**示例：**

#include <fstream>  
#include <string>  
void test01()  
{  
 ifstream ifs;  
 ifs.open("test.txt", ios::in);  
  
 if (!ifs.is\_open())  
 {  
 cout << "文件打开失败" << endl;  
 return;  
 }  
  
 //第一种方式  
 //char buf[1024] = { 0 };  
 //while (ifs >> buf)  
 //{  
 // cout << buf << endl;  
 //}  
  
 //第二种  
 //char buf[1024] = { 0 };  
 //while (ifs.getline(buf,sizeof(buf)))  
 //{  
 // cout << buf << endl;  
 //}  
  
 //第三种  
 //string buf;  
 //while (getline(ifs, buf))  
 //{  
 // cout << buf << endl;  
 //}  
  
 char c;  
 while ((c = ifs.get()) != EOF)  
 {  
 cout << c;  
 }  
  
 ifs.close();  
  
  
}  
  
int main() {  
  
 test01();  
  
 system("pause");  
  
 return 0;  
}

总结：

* 读文件可以利用 ifstream ，或者fstream类
* 利用is\_open函数可以判断文件是否打开成功
* close 关闭文件

## 二进制文件

* 以二进制的方式对文件进行读写操作 (好处在于可以打开自定义类型文件)，但是有可能会导致乱码
* 打开方式要指定为 ios::binary

### 写文件

* 二进制方式写文件主要利用流对象调用成员函数write
* 函数原型 ：ostream& write(const char \* buffer,int len);

参数解释：字符指针buffer指向内存中一段存储空间。len是读写的字节数

**示例：**

#include <fstream>  
#include <string>  
  
class Person  
{  
public:  
 char m\_Name[64];  
 int m\_Age;  
};  
  
//二进制文件 写文件  
void test01()  
{  
 //1、包含头文件  
  
 //2、创建输出流对象  
 ofstream ofs("person.txt", ios::out | ios::binary);  
   
 //3、打开文件  
 //ofs.open("person.txt", ios::out | ios::binary);  
  
 Person p = {"张三" , 18};  
  
 //4、写文件  
 ofs.write((const char \*)&p, sizeof(p));  
  
 //5、关闭文件  
 ofs.close();  
}  
  
int main() {  
  
 test01();  
  
 system("pause");  
  
 return 0;  
}

总结：

* 文件输出流对象 可以通过write函数，以二进制方式写数据

### 读文件

* 二进制方式读文件主要利用流对象调用成员函数read
* 函数原型：istream& read(char \*buffer,int len);

参数解释：字符指针buffer指向内存中一段存储空间。len是读写的字节数

示例：

#include <fstream>  
#include <string>  
  
class Person  
{  
public:  
 char m\_Name[64];  
 int m\_Age;  
};  
  
void test01()  
{  
 ifstream ifs("person.txt", ios::in | ios::binary);  
 if (!ifs.is\_open())  
 {  
 cout << "文件打开失败" << endl;  
 }  
  
 Person p;  
 ifs.read((char \*)&p, sizeof(p));  
  
 cout << "姓名： " << p.m\_Name << " 年龄： " << p.m\_Age << endl;  
}  
  
int main() {  
  
 test01();  
  
 system("pause");  
  
 return 0;  
}

* 文件输入流对象 可以通过read函数，以二进制方式读数据

注意： 对文件进行操作时，字符串建议使用C 语言的字符数组（char strs [64] ），不要使用C++ 的字符串（string strs ;）