# 一

共3个CSV文件

CsvReader

CsvFile

CsvWriter

# 二

先熟悉三个概念

class CsvRecord:

List<string> Fields

int FieldCount

CsvReader类代表Csv文件的一行，也就是一条记录。Fields是个字符串数组，每个元素是记录的字段，数组的顺序也是Csv文件字段的顺序。

class CsvRecords(List<Record>):

pass

CsvRecords是记录的集合，类型是List<List<string>>.

这里有个技巧，组合可以用继承代替，是个骚操作。相当于为一个类起别名。如CsvRecord与List<List<CsvRecord>完全等价。

# 三 读取一个Csv文件

using (CsvReader reader = new CsvReader(@"C:\test.csv",Encoding.UTF8))

{

reader.HasHeaderRow = true; //这个属性用于设置读取的Csv文件是否有标题行。由程序员主动设置。

while (reader.ReadNextRecord())

{

Console.WriteLine(reader.Fields[0]);//记录的第一个字段

Console.WriteLine(reader.Fields[1]);//记录的第二个字段

}

}

## 注意点

必须使用using，它要释放文件资源。

CsvReader类是专门用来读一个Csv文件或Csv文件的流或Csv格式的字符串。

允许Csv文件记录的字段数目不一样。允许Csv文件有空行，此时Fields是个空字符串数组。

不要管HasHeader属性，不访问也不设置。

## CsvReader的实现思路

重要属性：List<string> Fields 重要方法：bool ReadNextRecord()

首先，CsvReader会有一个StreamReader获得Csv文件，Csv字符串。Fields初始值是null。

ReadNextRecord( )每运行一次，就会调用StreamReader读取一行记录并转化到属性Fields中。

当读取完最后一行后，ReadLine()返回null，ReadNextRecord()返回false。

# 四 生成一个Csv文件的内存结构模型

CsvFile属性：

List<string> Headers 标题

CsvRecords Records 记录集合

新建一个Csv文件内存模型的方法：

1. CsvFile csvFile = new CsvFile(); //此时Headers和Records都是空列表
2. csvFile. Populate(string filePath)//从一个Csv文件生成内存模型，此时Headers和Records已被填充。

我们现在可以修改或访问CsvFile, 获取某一行记录，获取某一行的某个字段：利用索引器

如何创建一个CsvFile？

CsvFile有两个属性Headers和Records，我们可以先CsvFile file = new CsvFile( ); 然后填充这两个暴露的属性。

# 五 在磁盘生成一个Csv文件

先创建一个CsvFile内存模型，再调用Write方法。记得用using。

using(CsvWriter writer = new CsvWriter())

{

CsvWriter.Write(CsvFile csvFile, Encoding encoding)

}