

跨平台的 AliceCSV 解析软件 使用手册

版本：V1.0

目录

一、软件概述.....	3
二、功能介绍与使用方法.....	4
1. 解析 CSV 文件内容为二维列表	4
2. 解析 CSV 文件的其中一行	4
3. 将表格输出为 CSV 文件	4
4. 修复 CSV 文件中出现的长度问题	5
5. 格式转换.....	6

一、软件概述

跨平台的 AliceCSV 解析软件是一款跨平台的对 CSV 文件进行操作的软件，可以把 CSV 解析为通用的二维列表或数组，简化了软件开发中对 CSV 文件操作的过程。软件还包含对 CSV 文件进行格式纠正和格式转换的功能。

软件具有兼容不规范的 CSV 文件的能力，针对用户在操作 CSV 文件时常见的错误进行了优化，在发现文件包含格式错误时能尽量还原作者原本的意图。

软件包含了 C++、Python 和 JavaScript 语言的实现，分别对应嵌入式软件 and 应用程序、数据处理和网页前端用途，可覆盖大多数开发用途。多种实现使得软件可以跨平台使用。

相比同类软件，跨平台的 AliceCSV 解析软件非常简洁，其中 C++ 实现除去注释和示例后全部代码仅 4476 字节，便提供了 CSV 文件的解析、转换等功能，有效减少了处理 CSV 文件所需的硬件资源，可用于嵌入式设备。

二、功能介绍与使用方法

1. 解析 CSV 文件内容为二维列表

本软件提供的 parseCSV 函数用于解析 CSV 文件的内容。

以 Python 版本为例，读取解析的 CSV 文件的文本内容，传入 parseCSV 函数，即可得到解析为二维列表的表格。

```
Python 3.12.4 (tags/v3.12.4:8e8a4ba, Jun 6 2024, 19:30:16) [MSC v.1940 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import aliceCSV
>>> myFile = open("test.csv", encoding="utf-8")
>>> myTable = aliceCSV.parseCSV(myFile.read())
>>> print(myTable)
[['name', 'gender', 'height', 'address'], ['John', 'male', '175cm', '123 Main Street, New York, USA'], ['Emily', 'female', '160cm', '45 Oxford Road, London, UK'], ['Michael', 'male', '180cm', '10 Rue de la Paix, Paris, France'], ['Sophia', 'female', '165cm', '25 Alexanderplatz, Berlin, Germany'], ['']]
>>> |
```

函数需要的参数和对应的含义如下：

parseCSV(csv_text, [可选]delimiter)

csv_text: 要解析的 CSV 文件文本。

delimiter: CSV 文件的分隔符，可以不填，默认为","。

2. 解析 CSV 文件的其中一行

用户可使用软件的 parseLine 函数来解析 CSV 文件的单独某一行。

parseLine(line, delimiter):

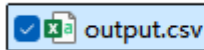
line: CSV 文件某一行的文本。

delimiter: 解析时选择的分隔符。可以不填，默认为","。

3. 将表格输出为 CSV 文件

writeCSV 函数可以将二维列表表示的表格保存为 CSV 文件。

```
>>> myTable = [{"用户名", "年龄", "身高"},
...             ["张三", "55", "177"],
...             ["李四", "67", "161"]]
>>>
>>> import aliceCSV
>>> aliceCSV.writeCSV(myTable)
>>> |
```



注意，由于各编程语言的 IO 逻辑不同，不同实现存在一定差异。例如，JavaScript 实现的 writeCSV 函数返回的是一个 blob 形式的 CSV 文件，而不是直接写入到硬盘。

函数需要的参数和对应的含义如下：

```
writeCSV(sheet, [可选]output_path, [可选]delimiter,
            [可选]sheet_encoding, [可选]line_break)
```

sheet: 需要保存的二维列表。

output_path: 输出路径。可以不填，默认为在当前目录下创建"output.csv"。

sheet_encoding: 输出文件的编码格式。可以不填，默认为"utf-8"。

delimiter: CSV 文件所使用的分隔符。可以不填，默认为","。

line_break: 输出文件使用的换行符。可以不填，默认为"\n"。

4. 修复 CSV 文件中出现的长度问题

由于种种原因，一些 CSV 文件每行的字段数不一样，这不符合国际上常用的 RFC 4180 规范，在一些场景下可能会引发问题。用户可以使用软件的 fixLineLength 函数使得每行字段数相同。

例如，下面这个表格各行长度不同。

姓名	年龄	职业		
张三	30	工程师		
李四	25			
王五	35	设计师	爱好:篮球	
赵六	40	教师	居住地:北京	电话:1234567890

我们可以使用 fixLineLength 进行修复

```
>>> import alic CSV
>>> myFile = open("error.csv", encoding="utf-8")
>>> myTable = alic CSV.parseCSV(myFile.read())
>>> alic CSV.fixLineLength(myTable)
[['姓名', '年龄', '职业', '', ''], ['张三', '30', '工程师', '', ''], ['李四', '25', '', '', ''], ['王五', '35', '设计师', '爱好:篮球', ''], ['赵六', '40', '教师', '居住地:北京', '电话:1234567890']]
```

把结果保存为 CSV 文件再打开，可以看到表格的每一行长度都变为了 5。

姓名	年龄	职业		
张三	30	工程师		
李四	25			
王五	35	设计师	爱好:篮球	
赵六	40	教师	居住地:北京	电话:1234567890

函数需要的参数和对应的含义如下：

`fixLineLength(csv_sheet)`

`csv_sheet`: 二维列表表示的表格。

5. 格式转换

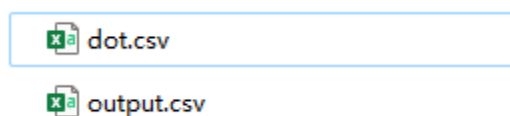
软件的 `fixCSV` 函数可以把 CSV 文件保存为各种兼容格式的 CSV 文件，包括改为使用其他分隔符、文件编码、换行符等。

例如，对于一个分隔符是“.”的 CSV 文件，可以使用 `fixCSV` 函数把它转换为常见的以逗号为分隔符的 CSV 文件。

```
用户名.年龄.身高
张三.55.177
李四.67.161
```

如图，使用 Python 实现的 `fixCSV` 函数，输入源文件地址和源文件分隔符，即可在当前路径输出转换成的“output.csv”。

```
>>> import aliceCSV
>>> aliceCSV.fixCSV("dot.csv", origin_delimiter=".")
>>> |
```



The image shows a file explorer interface with two files listed: 'dot.csv' and 'output.csv'. Both files have a green icon with a white 'x' and a blue outline, indicating they are CSV files. The 'dot.csv' file is selected, and the 'output.csv' file is listed below it.

注意，由于各编程语言 IO 操作的逻辑不同，不同实现会有一定差异。在 JavaScript 实现中，`fixCSV` 函数返回的是一个 **Promise**，用户解析这个 **Promise** 即可得到 **blob** 形式的转换后的文件。

可以根据实际需要输入更多参数。函数需要的参数和对应的含义如下：

`fixCSV(path, [可选]output_path, [可选]origin_delimiter, [可选]target_delimiter, [可选]origin_encoding)`

`path`: 输入的初始 CSV 文件路径。

`output_path`: 输出生成的 CSV 文件的路径。可以不填，默认为“output.csv”。JavaScript 实现没有这一参数。

`origin_delimiter`: 原始 CSV 文件的分隔符。可以省略，默认值为“.”。

`target_delimiter`: 输出文件中使用的分隔符。可以省略，默认值为“,”。

`origin_encoding`: 原始文件的编码。可以省略，默认值为“utf-8”。

`target_encoding`: 输出文件中使用的编码。可以省略，默认值为“utf-8”。

`target_line_break`: 输出文件的换行符。可以省略，默认值为“\n”。