> *Wiki* ▸ *[[API--中文手册]]* ▸ *[[核心函数]]* ▸ *CSV格式化*

- 如发现翻译不当或有其他问题可以通过以下方式联系译者:
- 邮箱：zhang_tianxu@sina.com
- QQ群：D3数据可视化205076374，大数据可视化436442115
- QQ群：D3数据可视化205076374，大数据可视化436442115

CSV 格式化 (d3.csv)

- `d3.csv - 获取一个CSV (comma-separated values, 冒号分隔值)文件。`
  - `d3.csv.parse - 将CSV文件字符串转化成object的数组，object的key由第一行决定。如: [{"Year": "1997", "Length": "2.34"}, {"Year": "2000", "Length": "2.38"}]`
    - `d3.csv.parseRows - 将CSV文件字符串转化成数组的数组。如: [ ["Year", "Length"],["1997", "2.34"],["2000", "2.38"] ]`
      - `d3.csv.format - 将object的数组转化成CSV文件字符串，是d3.csv.parse的逆操作。`
        - `d3.csv.formatRows - 将数组的数组转化成CSV文件字符串，是d3.csv.parseRows的逆操作。`
          - `d3.tsv - 获取一个TSV (tab-separated values, tab分隔值)文件。`
            - `d3.tsv.parse - 类似于d3.csv.parse。`
              - `d3.tsv.parseRows - 类似于d3.csv.parseRows。`
                - `d3.tsv.format - 类似于d3.csv.format。`
                  - `d3.tsv.formatRows - 类似于d3.csv.formatRows。`
                    - `d3.dsv - 创建一个类似于d3.csv的文件处理对象，可以自定义分隔符和mime type。如: vardsv = d3.dsv("|",`

```
"text/plain");
```

D3对逗号分隔值(CSV, comma-separated values)，制表符分隔值(TSV, tab-separated values)和任意分隔值(DSV, arbitrary delimiter-separated values)等数据格式的解析提供内置支持。这些表格格式深受诸如Microsoft Excel电子表格程序。表格格式通常比JSON更省空间,这可以提高大型数据集加载时间。

d3.**csv**(*url*[[, *accessor*], *callback*])

在指定的*url*为逗号分隔值(CSV)文件发出一个HTTP GET请求。一般认为文件内容是RFC4180-compliant。请求的mime（多用途互联网邮件扩展类型）类型一般为“text/csv”。此请求会被异步处理,在打开请求后此方法会立即返回。CSV数据可用时,将以parsed rows作为参数调用指定的回调。如果出现错误,此回调函数将返回null。一个可选的访问器方法可能被指定,然后传递给d3.csv.parse;也可以通过使用返回请求对象的row函数来指定。例如:

```
d3.csv("path/to/file.csv")
    .row(function(d) { return {key: d.key, value: +d.value}; })
    .get(function(error, rows) { console.log(rows); });
```

查看样例：unemployment choropleth。

d3.csv.**parse**(*string*[, *accessor*])

通过一个CSV文件的内容解析指定的字符串，返回一个代表解析行的对象数组。一般认为文件内容是RFC4180-compliant。与parseRows方法不同的是,这种方法要求CSV文件的第一行包含一个以逗号分隔的列名;这些列名成为返回的对象的属性。例如,参考以下CSV文件:

```
Year,Make,Model,Length
1997,Ford,E350,2.34
2000,Mercury,Cougar,2.38
```

生成的JavaScript数组:

```
[
  {"Year": "1997", "Make": "Ford", "Model": "E350", "Length": "2.34"},
  {"Year": "2000", "Make": "Mercury", "Model": "Cougar", "Length": "2.38"}
]
```

值得注意的是这些值都是字符串；它们不会自动转为数字类型值。JavaScript会强制字符串自动转换成数字类型值(例如,使用+运算符)。通过指定一个访问器函数,您可以将字符串转换为数字或其他特定的类型,如日期:

```
d3.csv("example.csv", function(d) {
  return {
    year: new Date(+d.Year, 0, 1), // convert "Year" column to Date
    make: d.Make,
    model: d.Model,
    length: +d.Length // convert "Length" column to number
  };
}, function(error, rows) {
```

```
  console.log(rows);
});
```

尽管由很多的限制，但使用连接符"+"比parseInt或parseFloat 通常更快。例如,"30 px"当强制使用"+"返回NaN,而parseInt和parseFloat返回30。

d3.csv.**parseRows**(*string*[, *accessor*])

通过一个CSV文件的内容解析指定的字符串，返回一个代表解析行的对象数组。一般认为文件内容是RFC4180-compliant。与parse 方法,不论CSV文件是否不包含一个头，该方法将标题行作为标准并且使用。每一行都被表示为一个数组而不是一个对象。行可能会变长。例如,考虑以下CSV文件:

```
1997,Ford,E350,2.34
2000,Mercury,Cougar,2.38
```

生成的JavaScript数组:

```
[
  ["1997", "Ford", "E350", "2.34"],
  ["2000", "Mercury", "Cougar", "2.38"]
]
```

值得注意的是这些值都是字符串；它们不会自动转为数字类型值。有关详细请参阅parse。 第二个参数（[, accessor]）可以指定一个访问器函数。这个函数调用CSV文件中的每一行数据，通过当前行数据对象和当前行索引作为两个参数。函数的返回值将取代所在返回数组里的元素数据；如果函数返回null，此行便从返回的数组里剔除。实际上，这个访问器类似于map 和filter操作符去返回数据行。访问器函数通过parse将每一行转换为一个带有一些已命名属性的对象。

d3.csv.**format**(*rows*)

将指定数组里的行内容转换为逗号分隔值格式的字符串并返回。这个操作是parse方法的逆转。每一行将会由一个换行符(\ n)隔开,并在每一行的每一列将以逗号(,)隔开。数据值中包含的逗号，双引号(")或换行符会使用双引号将其取代（最后这句实践后不理解，理解不透）。

每一行视为一个对象,并且所有的对象属性将被转换成字段。为了更好的控制那些被转换的属性，将行内容转换为只包含这些应该被转换的属性的数组并且使用formatRows方法。

d3.csv.**formatRows**(*rows*)

将指定数组里的行内容转换为逗号分隔值格式的字符串并返回。这个操作是parseRows方法的逆转。每一行将会由一个换行符(\ n)隔开,并在每一行的每一列将以逗号(,)隔开。值所包含的逗号,双引号(")或换行符会使用双引号将其脱逃。

TSV

除了分隔符由制表符代替了逗号（制表符分隔值相当于逗号分隔值）,其它没有太大区别。

d3.**tsv**(*url*[, *accessor*][, *callback*])

相当于d3.csv，只是分隔符为制表符而已。

d3.tsv.**parse**(*string*[, *accessor*])

相当于csv.parse，只是分隔符为制表符而已。

d3.tsv.**parseRows**(*string*[, *accessor*])

相当于csv.parseRows，只是分隔符为制表符而已。

d3.tsv.**format**(*rows*)

相当于csv.format，只是分隔符为制表符而已。

d3.tsv.**formatRows**(*rows*)

相当于csv.formatRows，只是分隔符为制表符而已。

Arbitrary Delimiters

d3.**dsv**(*delimiter*, *mimeType*)

对于给定分隔符和mime类型构造一个新的解析器。例如,解析值由"|"分隔,竖线字符,使用如下:

```
var dsv = d3.dsv("|", "text/plain");
```

dsv(*url*[, *accessor*][, *callback*])

相当于d3.csv，只是分隔符为具体值而已。

dsv.**parse**(*string*[, *accessor*])

相当于csv.parse，只是分隔符为具体值而已。

dsv.**parseRows**(*string*[, *accessor*])

相当于csv.parseRows，只是分隔符为具体值而已。

dsv.**format**(*rows*)

相当于csv.format，只是分隔符为具体值而已。

dsv.**formatRows**(*rows*)

相当于csv.formatRows，只是分隔符为具体值而已。

HarryT20140329

MIME: https://zh.wikipedia.org/wiki/多用途互聯網郵件擴展 (Howard L. -- 11.Jan.2016)