

Quick links: [Flags](#) [Verbs](#) [Functions](#) [Glossary](#) [Release docs](#)

# Introduction

Miller is a command-line tool for querying, shaping, and reformatting data files in various formats, including CSV, TSV, JSON, and JSON Lines.

**The big picture:** Even well into the 21st century, our world is full of text-formatted data such as CSV. Google CSV *memes*, for example. We need tooling to *thrive in this world*, nimbly manipulating data which is in CSVs. And we need tooling to *move beyond CSV*, to be able to pull data out and into other storage and processing systems. Miller is designed for both of these goals.

In several senses, Miller is more than one tool:

**Format conversion:** You can convert CSV files to JSON, or vice versa, or pretty-print your data horizontally or vertically to make it easier to read.

**Data manipulation:** With a few keystrokes, you can remove columns you don't care about -- or make new ones.

**Pre-processing/post-processing vs standalone use:** You can use Miller to clean data files and put them into standard formats, perhaps in preparation for loading them into a database or a hands-off data-processing pipeline. Or you can use it post-process and summarize database-query output. As well, you can use Miller to explore and analyze your data interactively.

**Compact verbs vs programming language:** For low-keystroking, you can do things like

```
mlr --csv sort -f name input.csv
```

```
mlr --json head -n 1 myfile.json
```

The `sort`, `head`, etc., are called *verbs*. They're analogs of familiar command-line tools like `sort`, `head`, and so on -- but they're aware of name-indexed, multi-line file formats like CSV, TSV, and JSON. In addition, though, using Miller's `put` verb, you can use programming-language statements for expressions like

```
mlr --csv put '$rate = $units / $seconds' input.csv
```

which allow you to express your own logic succinctly.

**Multiple domains:** People use Miller for data analysis, data science, software engineering, devops/system-administration, journalism, scientific research, and more.

In the following, you can see how CSV, TSV, tabular, JSON, and other **file formats** share a common theme which is **lists of key-value-pairs**. Miller embraces this common theme.

```
$ cat example.csv
color,shape,flag,index,quantity,rate
yellow,triangle,1,11,43.6498,9.8870
red,square,1,15,79.2778,0.0130
red,circle,1,16,13.8103,2.9010
red,square,0,48,77.5542,7.4670
purple,triangle,0,51,81.2290,8.5910
red,square,0,64,77.1991,9.5310
purple,triangle,0,65,80.1405,5.8240
yellow,circle,1,73,63.9785,4.2370
yellow,circle,1,87,63.5058,8.3350
purple,square,0,91,72.3735,8.2430
```

```
$ mlr --icsv --oprint sort -f color,shape example.csv
color shape flag index quantity rate
purple square 0 91 72.3735 8.2430
purple triangle 0 51 81.2290 8.5910
purple triangle 0 65 80.1405 5.8240
red circle 1 16 13.8103 2.9010
red square 1 15 79.2778 0.0130
red square 0 48 77.5542 7.4670
red square 0 64 77.1991 9.5310
yellow circle 1 73 63.9785 4.2370
yellow circle 1 87 63.5058 8.3350
yellow triangle 1 11 43.6498 9.8870
```

```
$ mlr --icsv --ojson filter '$color=="yellow"' example.csv
{
  "color": "yellow",
  "shape": "triangle",
  "flag": 1,
  "index": 11,
  "quantity": 43.6498,
  "rate": 9.8870
}
{
  "color": "yellow",
  "shape": "circle",
  "flag": 1,
  "index": 73,
  "quantity": 63.9785,
  "rate": 4.2370
}
{
  "color": "yellow",
  "shape": "circle",
  "flag": 1,
  "index": 87,
  "quantity": 63.5058,
  "rate": 8.3350
}
```

```
$ mlr --c2p --from example.csv put '$qr = $quantity * $rate'
color shape flag k index quantity rate qr
yellow triangle true 1 11 43.6498 9.8870 431.5655726
red square true 2 15 79.2778 0.0130 1.0306114
red circle true 3 16 13.8103 2.9010 40.063680299999994
red square false 4 48 77.5542 7.4670 579.0972113999999
purple triangle false 5 51 81.2290 8.5910 697.8383389999999
red square false 6 64 77.1991 9.5310 735.7846221000001
purple triangle false 7 65 80.1405 5.8240 466.738272
yellow circle true 8 73 63.9785 4.2370 271.0769045
yellow circle true 9 87 63.5058 8.3350 529.3208430000001
purple square false 10 91 72.3735 8.2430 596.5747605000001
```