CSV Rows Into Objects

A CSV loader is Chapters 5, 7, and 14 meeting at once -- split the row, build the object, collect the list. The payoff is data.getPassword() instead of data[1].

Automation Foundations · ashishautomationlabs.store

Here is where several chapters meet. A CSV file holds test data; Chapter 14 splits each row; Chapter 7 turns the fields into an object; Chapter 5 collects them into a list:

```
try (BufferedReader reader = Files.newBufferedReader(file)) {
    String header = reader.readLine();
    String line;
    while ((line = reader.readLine()) != null) {
        if (line.isBlank()) {
            continue;
        }
        String[] parts = line.split(",", -1);
        rows.add(new LoginData(parts[0], parts[1], parts[2]));
    }
}
```

```
Header skipped: username,password,expected
Rows loaded   : 2
  priya / Valid#123 -> expect PASS
  raj / wrong -> expect FAIL
```

Read the loop condition, because it is a common Java idiom worth recognizing: `(line = reader.readLine()) != null` reads the next line, assigns it, and checks whether it exists -- `readLine` returns `null` at the end of the file. This is the one place where assignment inside a condition is normal and correct, which is worth noting after Chapter 3 taught you to fear it.

Four details make this production-worthy rather than a demonstration. The header line is read once before the loop, so it is skipped rather than parsed as data. Blank lines are skipped with `continue` from Chapter 4, because real files end with one. `split(",", -1)` keeps trailing empty fields, which matters when a row ends with an empty column and your field positions must line up. And the result is a `List<LoginData>` -- real objects with names, not a `String[]` you index by number and hope. That last point is the difference between `data[1]` and `data.getPassword()`. Six months later, only one of them is readable.

> *Using the CSV loading code above: why is `reader.readLine()` called once before the loop starts? What does `readLine()` return at the end of the file, and how does the loop use that? Why `split(",", -1)` instead of `split(",")`?*

>

> *Answer -- to consume the header row so it is not treated as data.* `null`*, which is what ends the* `while` *loop. And the* `-1` *keeps trailing empty fields, so a row ending in an empty column still produces the full set of positions.*

Data flows both ways. Loading a CSV into objects is one direction -- writing results back out is the same pattern in reverse, and it's where this chapter's most important rule lives.