See also

- The *Using awk for advanced text processing* recipe in this chapter explains the `awk` command
- The *Arrays and associative arrays* recipe in *Chapter 1*, *Shell Something Out*, explains the arrays in Bash

Compressing or decompressing JavaScript

JavaScript is widely used for designing websites. While writing the JavaScript code, we use several white spaces, comments, and tabs for readability and maintenance of the code. This causes the file size to increase, and as the file size increases, so does the time taken to load the page. Hence, most of the professional websites use compressed JavaScript for fast loading. This compression (also known as **minified JS**) is mostly squeezing white spaces and newline characters. Once JavaScript is compressed, it can be decompressed by adding enough white space and newline characters, which makes it readable. This recipe is an attempt to produce similar capabilities in the shell.

Getting ready

We are going to write a JavaScript compressor or obfuscation tool as well as a decompressing tool. Let's consider the following sample JavaScript:

```
$ cat sample.js
function sign_out()
{

  $("#loading").show();
  $.get("log_in",{logout:"True"},

  function(){
    window.location="";
  });
}
```

Following are the tasks that we need to perform for compressing this JavaScript:

1. Remove newline and tab characters.
2. Squeeze spaces.
3. Replace comments that look like `/* content */`.

To decompress or to make the JavaScript more readable, we can use the following tasks:

- Replace `;` with `;\n`
- Replace `{` with `{\n`, and `}` with `\n}`

How to do it...

Using the steps that we now have in mind, we can use the following command chain:

```
$ cat sample.js |  \
tr -d '\n\t' |  tr -s ' ' \
| sed 's:/\*.*\*/::g' \
| sed 's/ \?\([{}();,:]\) \?/\1/g'
```

The output is as follows:

```
function sign_out(){$("#loading").show();$.get("log_
in",{logout:"True"},function(){window.location="";});}
```

We can write a decompression script for making the obfuscated code readable, as follows:

```
$ cat obfuscated.txt | sed 's/;/;\n/g; s/{/{\n\n/g; s/}/\n\n}/g'
```

Or:

```
$ cat obfuscated.txt | sed 's/;/;\n/g' | sed 's/{/{\n\n/g' | sed 's/}/\n\
n}/g'
```

There is a limitation in the script that it even gets rid of extra spaces where it is intended. For example, if you have a line like the following:

```
var a = "hello world"
```

The two spaces will be converted into one space. Things like these are possible to fix by using the pattern-matching tools we have discussed. Also, when dealing with mission-critical JavaScript code, it is advised to use well-established tools to do this.

How it works...

The compression command performs the following tasks:

- Removes the `\n` and `\t` characters:

```
tr -d '\n\t'
```