

在 web 方面，有三个压缩方式

- gzip
- deflate
- brotli

历史

现在所有的无损压缩算法基本上都可以追溯到两个压缩算法 LZ77 和 LZ78

LZ77 算法非常简单，它将文本当中一些重复的地方用三元组 $\langle distance, length, character \rangle$ 表示

```
document.addEventListener("click", f);  
  
document.addEventListener("click", g);
```

看上面这段 JavaScript 代码，LZ77 会怎么处理呢，它将 `document.addEventListener("click",` 这部分用一个元组替代

```
document.addEventListener("click", f);  
  
<13, 32> g);
```

这里的元组代表什么意思呢？这里的 13 代表向前回退 13 字节，32 代表复制 32 个字节

肉眼可见，整段文本被缩小了，但是有一个很明显的问题是，所谓文件内容在计算机当中都是一些字节，当完成压缩之后，怎么保证文件里面的元组是通过压缩算法生成的而非文件本身的内容如此，LZ77 用了一个显得有点呆瓜的方法，那就是将所有的文字内容通过元组形式表示，所以，假设有以下文本

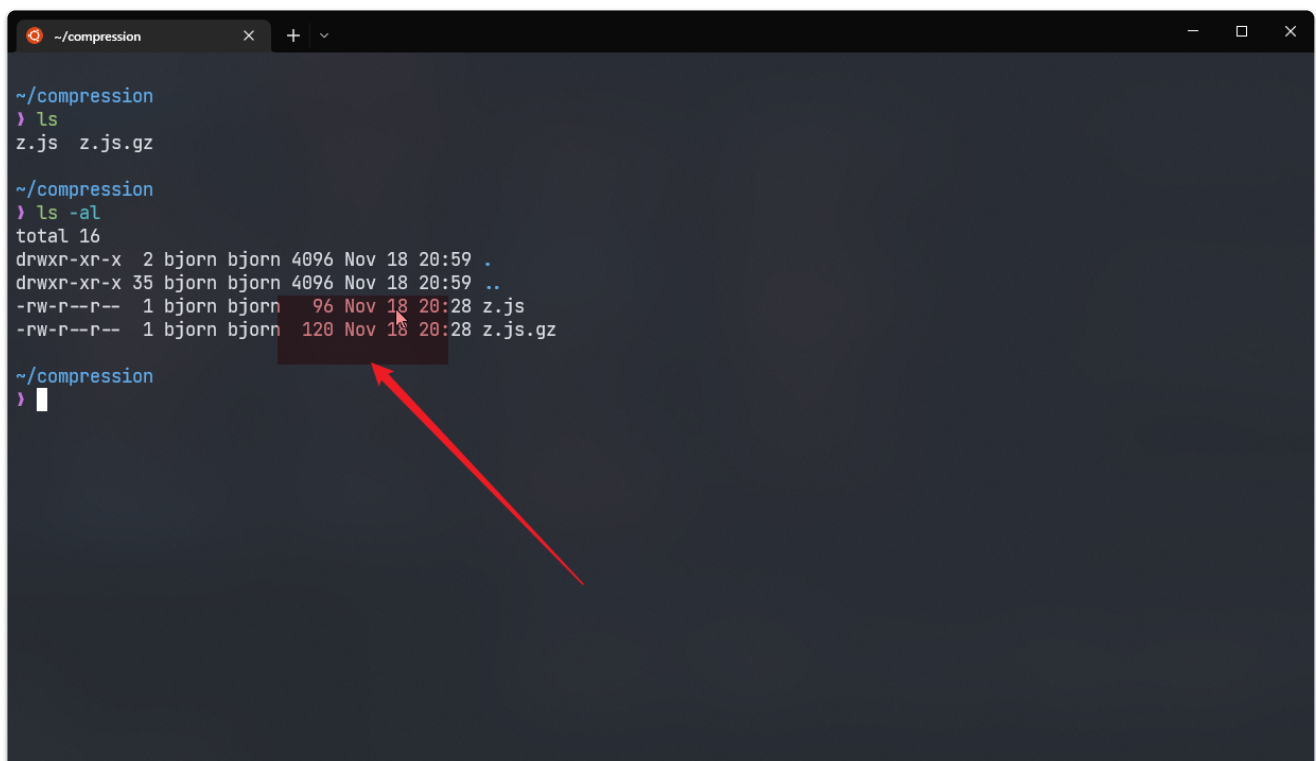
```
l d d
```

那么就会变成

```
<0,0,a><0,0,b><0,0,c><1,1,E>
```

从以前的一个字节，就会变成四个字节，这里<0,0,a>代表原来的 a，通过这样的三元组来代表原来的整个文本文件内容。

这样直接的方式，肯定在某种情况下会出现不仅不会压缩文件，而且文件本身还会变大的滑稽情况



A terminal window titled ~/compression showing the results of file compression. The first command 'ls' shows files z.js and z.js.gz. The second command 'ls -al' shows detailed file information. A red arrow points to the file sizes: z.js is 96 bytes and z.js.gz is 120 bytes, indicating that the compressed file is larger than the original.

```
~/compression
) ls
z.js  z.js.gz

~/compression
) ls -al
total 16
drwxr-xr-x  2 bjorn bjorn 4096 Nov 18 20:59 .
drwxr-xr-x 35 bjorn bjorn 4096 Nov 18 20:59 ..
-rw-r--r--  1 bjorn bjorn  96 Nov 18 20:28 z.js
-rw-r--r--  1 bjorn bjorn 120 Nov 18 20:28 z.js.gz

~/compression
) 
```

大概五年之后，JACOB ZIV (LZ77 作者) 又提出了 LZSS 的

压缩算法，相比于 LZ77 算法，LZSS 即使在小文件的情况下也有良好的压缩率

gzip

言归正传，现在使用最多的压缩格式仍然是 GZIP，简单介绍一下 gzip 压缩，gzip 文件都以如下字节序列的格式开头

```
+---+---+---+---+---+---+---+---+---+
|ID1|ID2|CM |FLG|      MTIME      |XFL|OS | (more--
>)
+---+---+---+---+---+---+---+---+---+
```

也就是这一段

```
00000000: 1f8b 0808 e69a 7863 0003 6675 636b 2e74 .....xc..fuck.t
00000010: 7874 00cb 48cd c9c9 5728 2fca cf49 5148 xt..H...W(/..IQH
00000020: 2b4d ce56 28c9 4855 484f cde3 0200 fc7f +M.V(.HUHO.....
00000030: 7b31 1900 0000                               {1....
```

- ID1-ID2 表示这是 gzip 格式的文件
- CM (Compression Method) 表示使用的压缩方法，0-7 都是保留位，8 代表使用 **defalte**
- FLG 表示 header 之后的字节表示啥
- MTIME 表示文件压缩完成之后的时间，
- XFL...
- OS...

重点在 CM 这个字节，他表示压缩的方法 deflate，也就是实际上 gzip 使用仍然是 deflate 相同的压缩算法，但是仍然有一些不同的地方

| gzip | defalte |

| ----- | ----- |
| crc32/crc16 | alter-32 |
| | |
| | |



网络传输当中如何使用压缩

想要使用压缩，必须客户端支持，浏览器通过指定请求头中的

```
Accept-encoding: gzip br deflate
```

来表明支持gzip brotli deflate格式的压缩算法

当请求发送给服务器之后，服务器最终决定使用哪种压缩算法，然后通过指定响应头中的

```
Content-Encoding: gzip
```

来告诉浏览器：“hey，我们使用 gzip 吧 🤗”

当然 Content-Encoding 能够指定的值不只是上面提到的三个，能够使用的选项在 [Hypertext Transfer Protocol \(HTTP\) Parameters \(iana.org\)](https://tools.ietf.org/html/rfc7231#section-4.1) 能够找到