link: null
title: 珠峰架构师成长计划
description: buf.fill(value[, offset[, end]][, encoding])
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=50 sentences=101, words=562

1. 什么是Buffer

- 缓冲区Buffer是暂时存放输入输出数据的一段内存。
- JS语言没有二进制数据类型，而在处理TCP和文件流的时候，必须要处理二进制数据。
- NodeJS提供了一个Buffer对象来提供对二进制数据的操作
- 是一个表示固定内存分配的全局对象，也就是说要放到缓存区中的字节数需要提前确定
- Buffer好比由一个8位字节元素组成的数组，可以有效的在JavasScript中表示二进制数据

2. 什么是字节

- 字节(Byte)是计算机存储时的一种计量单位，一个字节等于8位二进制数
- 一个位就代表一个0或1，每8个位（bit）组成一个字节（Byte）
- 字节是通过网络传输信息的基本单位
- 一个字节最大值十进制数是255(2**8-1)

3. 进制

- 0b 2进制
- 0x 16进制
- 0o 8进制
- 将任意进制字符串转换为十进制

```
parseInt("11", 2);
parseInt("77", 8);
parseInt("e7", 16);
```

- 将10进制转换为其它进制字符串

```
(3).toString(2)
(17).toString(16)
(33).toString(32)
```

4. 定义buffer的三种方式

```
const buf1 = Buffer.alloc(10);

const buf2 = Buffer.alloc(10, 1);

const buf3 = Buffer.allocUnsafe(10);
```

```
const buf4 = Buffer.from([1, 2, 3]);
```

> 正常情况下为0-255之间;

```
const buf5 = Buffer.from('珠峰培训');
```

5.buffer常用方法

buf.fill(value[, offset[, end]][, encoding])

```
buffer.fill(0);
```

buf.write(string[, offset[, length]][, encoding])

```
let buffer = Buffer.allocUnsafe(6);
buffer.write('珠',0,3,'utf8');
buffer.write('峰',3,3,'utf8');
```

- 通过指定的 offset 将 value 写入到当前 Buffer 中。
- 这个 value 应当是一个有效的有符号的8位整数

buf.writeInt8(value, offset[, noAssert])

```
let buf = Buffer.alloc(4);
buf.writeInt8()
buf.writeInt8(0,0);
buf.writeInt8(16,1);
buf.writeInt8(32,2);
buf.writeInt8(48,3);
console.log(buf);
console.log(buf.readInt8(0));
console.log(buf.readInt8(1));
console.log(buf.readInt8(2));
console.log(buf.readInt8(3));
```

不同的CPU有不同的字节序类型，这些字节序是指整数在内存中保存的顺序。

- Big-endian：将高序字节存储在起始地址（高位编址）
- Little-endian：将低序字节存储在起始地址（低位编址）

```
let buffer = Buffer.alloc(4);
buffer.writeInt16BE(2**8,0);
console.log(buffer);
console.log(buffer.readInt16BE(0));

buffer.writeInt16LE(2**8,2);
console.log(buffer);
console.log(buffer.readInt16LE(2));
```

```
buf.toString([encoding[, start[, end]]])
```

```
let buffer = Buffer.from('珠峰架构');
console.log(buffer.toString('utf8',3,6));
```

```
buf.slice([start[, end]])
```

```
let buffer = Buffer.from('珠峰架构');
let subBuffer = buffer.slice(0,6);
console.log(subBuffer.toString());
```

```
let {StringDecoder}  = require('string_decoder');
let sd = new StringDecoder();
let buffer = Buffer.from('珠峰');
console.log(sd.write(buffer.slice(0,4)));
console.log(sd.write(buffer.slice(4)));
```

- 复制Buffer 把多个buffer拷贝到一个大buffer上

```
buf.copy(target[, targetStart[, sourceStart[, sourceEnd]]])
```

```
let buffer = Buffer.from('珠峰架构');
let subBuffer = Buffer.alloc(6);
buffer.copy(subBuffer,0,0,4);
buffer.copy(subBuffer,3,3,6);
console.log(subBuffer.toString());
```

```
Buffer.prototype.copy = function(targetBuffer,targetStart,sourceStart,sourceEnd){
    for(let i=sourceStart;ithis[i];
    }
}
let buffer = Buffer.from('珠峰');
let subBuffer = Buffer.alloc(6);
buffer.copy(subBuffer,0,0,4);
buffer.copy(subBuffer,3,3,6);
console.log(subBuffer.toString());
```

```
Buffer.concat(list[, totalLength])
```

```
let buffer1 = Buffer.from('珠');
let buffer2 = Buffer.from('峰');
let buffer = Buffer.concat([buffer1,buffer2]);
console.log(buffer.toString());
```

```
Buffer.concat = function (list) {
    let totalLength = list.reduce((len, item) => len + item.length, 0);
    if (list.length == 0)
        return list[0];
    let newBuffer = Buffer.alloc(totalLength);
    let pos = 0;
    for (let buffer of list) {
        for (let byte of buffer) {
            newBuffer[pos++] = byte;
        }
    }
    return newBuffer;
}
let buffer1 = Buffer.from('珠');
let buffer2 = Buffer.from('峰');
let buffer = Buffer.concat([buffer1, buffer2]);
console.log(buffer.toString());
```

- 判断是否是buffer

```
Buffer.isBuffer();
```

- 获取字节长度(显示是字符串所代表buffer的长度)

```
let str = '珠峰';
console.log(str.length);
let buffer = Buffer.from(str);
console.log(Buffer.byteLength(buffer));
```

6. base64

- Base64是网络上最常见的用于传输8Bit字节码的编码方式之一
- Base64就是一种基于64个可打印字符来表示二进制数据的方法
- Base64要求把每三个8Bit的字节转换为四个6Bit的字节（3 _8 = 4_6 = 24），然后把6Bit再添两位高位0，组成四个8Bit的字节

```
const CHARTS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/';
function transfer(str){
  let buf = Buffer.from(str);
  let result = '';
  for(let b of buf){
      result += b.toString(2);
  }
  return result.match(/(\d{6})/g).map(val=>parseInt(val,2)).map(val=>CHARTS[val]).join('');
}
let r = transfer('珠');
console.log(r);
```