链滴

NODE 学习第三天

作者：cndinx

原文链接：https://ld246.com/article/1509936358463

来源网站：链滴

Node.js Buffer(缓冲区)

JavaScript 语言自身只有字符串数据类型，没有二进制数据类型。

但在处理像TCP流或文件流时，必须使用到二进制数据。因此在 Node.js中，定义了一个 Buffer 类，类用来创建一个专门存放二进制数据的缓存区。

在 Node.js 中，Buffer 类是随 Node 内核一起发布的核心库。Buffer 库为 Node.js 带来了一种存储始数据的方法，可以让 Node.js 处理二进制数据，每当需要在 Node.js 中处理I/O操作中移动的数据，就有可能使用 Buffer 库。原始数据存储在 Buffer 类的实例中。一个 Buffer 类似于一个整数数组但它对应于 V8 堆内存之外的一块原始内存。

创建 Buffer 类

Node Buffer 类可以通过多种方式来创建。

方法 1

创建长度为 10 字节的 Buffer 实例：

```
var buf = new Buffer(10);
```

方法 2

通过给定的数组创建 Buffer 实例：

```
var buf = new Buffer([10, 20, 30, 40, 50]);
```

方法 3

通过一个字符串来创建 Buffer 实例：

```
var buf = new Buffer("www.runoob.com", "utf-8");
```

utf-8 是默认的编码方式，此外它同样支持以下编码："ascii", "utf8", "utf16le", "ucs2", "base64" 和 hex"。

写入缓冲区

语法

写入 Node 缓冲区的语法如下所示：

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

参数

参数描述如下：

- **string** - 写入缓冲区的字符串。
- **offset** - 缓冲区开始写入的索引值，默认为 0 。
- **length** - 写入的字节数，默认为 buffer.length
- **encoding** - 使用的编码。默认为 'utf8' 。

返回值

返回实际写入的大小。如果 buffer 空间不足， 则只会写入部分字符串。

实例

```
buf =  new  Buffer(256);
len = buf.write("www.runoob.com");
console.log("写入字节数 : "+ len);
```

从缓冲区读取数据

语法

读取 Node 缓冲区数据的语法如下所示：

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

参数

参数描述如下：

- **encoding** - 使用的编码。默认为 'utf8' 。
- **start** - 指定开始读取的索引位置，默认为 0。
- **end** - 结束位置，默认为缓冲区的末尾。

返回值

解码缓冲区数据并使用指定的编码返回字符串。

实例

```
buf = new Buffer(26);
for (var i = 0 ; i < 26 ; i++) {
  buf[i] = i + 97;
}

console.log( buf.toString('ascii'));       // 输出: abcdefghijklmnopqrstuvwxyz
console.log( buf.toString('ascii',0,5));   // 输出: abcde
console.log( buf.toString('utf8',0,5));    // 输出: abcde
console.log( buf.toString(undefined,0,5)); // 使用 'utf8' 编码, 并输出: abcde
```

执行以上代码，输出结果为：

```
$ node main.js
abcdefghijklmnopqrstuvwxyz
abcde
abcde
abcde
```

将 Buffer 转换为 JSON 对象

语法

将 Node Buffer 转换为 JSON 对象的函数语法格式如下：

```
buf.toJSON()
```

返回值

返回 JSON 对象。

实例

```
var buf =  new  Buffer('www.runoob.com');
var json = buf.toJSON(buf);
console.log(json);
```

执行以上代码，输出结果为：

```
[  119,  119,  119,  46,  114,  117,  110,  111,  111,  98,  46,  99,  111,  109  ]
```

缓冲区合并

语法

Node 缓冲区合并的语法如下所示：

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

参数

参数描述如下：

- **list** - 用于合并的 Buffer 对象数组列表。
- **totalLength** - 指定合并后Buffer对象的总长度。

返回值

返回一个多个成员合并的新 Buffer 对象。

实例

```
var buffer1 = new Buffer('菜鸟教程 ');
var buffer2 = new Buffer('www.runoob.com');
var buffer3 = Buffer.concat([buffer1,buffer2]);
console.log("buffer3 内容: " + buffer3.toString());
```

执行以上代码，输出结果为：

```
buffer3 内容:  菜鸟教程 www.runoob.com
```

缓冲区比较

语法

Node Buffer 比较的函数语法如下所示, 该方法在 Node.js v0.12.2 版本引入：

```
buf.compare(otherBuffer);
```

参数

参数描述如下：

- **otherBuffer** - 与 **buf** 对象比较的另外一个 Buffer 对象。

返回值

返回一个数字，表示 **buf** 在 **otherBuffer** 之前，之后或相同。

实例

```
var buffer1 = new Buffer('ABC');
var buffer2 = new Buffer('ABCD');
var result = buffer1.compare(buffer2);

if(result < 0) {
   console.log(buffer1 + " 在 " + buffer2 + "之前");
}else if(result == 0){
   console.log(buffer1 + " 与 " + buffer2 + "相同");
}else {
   console.log(buffer1 + " 在 " + buffer2 + "之后");
}
```

执行以上代码，输出结果为：

```
ABC在ABCD之前
```

拷贝缓冲区

语法

Node 缓冲区拷贝语法如下所示：

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

参数

参数描述如下：

- **targetBuffer** - 要拷贝的 Buffer 对象。
- **targetStart** - 数字, 可选, 默认: 0
- **sourceStart** - 数字, 可选, 默认: 0
- **sourceEnd** - 数字, 可选, 默认: buffer.length

返回值

没有返回值。

实例

```
var buffer1 = new Buffer('ABC');
// 拷贝一个缓冲区
var buffer2 = new Buffer(3);
buffer1.copy(buffer2);
console.log("buffer2 content: " + buffer2.toString());
```

执行以上代码，输出结果为：

```
buffer2 content: ABC
```

缓冲区裁剪

Node 缓冲区裁剪语法如下所示：

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

参数

参数描述如下：

- **start** - 数字, 可选, 默认: 0
- **end** - 数字, 可选, 默认: buffer.length

返回值

返回一个新的缓冲区，它和旧缓冲区指向同一块内存，但是从索引 start 到 end 的位置剪切。

实例

```
var buffer1 = new Buffer('runoob');
// 剪切缓冲区
var buffer2 = buffer1.slice(0,2);
console.log("buffer2 content: " + buffer2.toString());
```

执行以上代码，输出结果为：

```
buffer2 content: ru
```

缓冲区长度

语法

Node 缓冲区长度计算语法如下所示：

```
buf.length;
```

返回值

返回 Buffer 对象所占据的内存长度。

实例

```
var buffer =  new  Buffer('www.runoob.com');
//  缓冲区长度
console.log("buffer length: "  + buffer.length);
```

执行以上代码，输出结果为：

```
buffer length:  14
```

转载出自：菜鸟教程