# safe-buffer

在所有版本的Node.js中使用新的Node.js缓冲区API（Buffer.from，Buffer.alloc，Buffer.allocUnsafe，Buffer.allocUnsafeSlow）。可用时使用内置的实现。

## install

npm install safe-buffer

## usage

这个包的目标是为node.js Buffer提供一个安全的替换。

这是Buffer的替代品。 您可以通过在node.js模块的顶部添加一个require行来使用它。

## api

### Class Method: Buffer.from(buffer)

* buffer {Buffer}

将传递的缓冲区数据复制到新的Buffer实例。

const buf1 = Buffer.from('buffer');

const buf2 = Buffer.from(buf1);

buf1[0] = 0x61;

console.log(buf1.toString());

  // 'auffer'

console.log(buf2.toString());

  // 'buffer' (copy is not changed)

如果缓冲区不是缓冲区，则会抛出TypeError。

### Class Method: Buffer.from(str[, encoding])

* str {String} String to encode.
* encoding {String} Encoding to use, Default: 'utf8'

创建一个包含给定JavaScript字符串str的新缓冲区。 如果提供，则编码参数标识字符编码。 如果没有提供，编码默认为'utf8'。

const buf1 = Buffer.from('this is a tést');

console.log(buf1.toString());

  // prints: this is a tést

console.log(buf1.toString('ascii'));

  // prints: this is a tC)st

const buf2 = Buffer.from('7468697320697320612074c3a97374', 'hex');

console.log(buf2.toString());

  // prints: this is a tést

如果str不是字符串，则会引发TypeError。

# etag

创建简单的HTTP ETags

这个模块生成HTTP ETags（在RFC 7232中定义）用于HTTP响应。

## Installation

$ npm install etag

## API

# fresh

HTTP响应新鲜度测试

## Installation

$ npm install fresh

## API

# ms

使用此软件包可以轻松地将各种时间格式转换为毫秒。

# parseurl

用memoization解析一个URL

## Install

$ npm install parseurl

## API

# path

resove方法：将参数 to 位置的字符解析到一个绝对路径里

path.resolve([from ...], to)

from                     源路径  
to                         将被解析到绝对路径的字符串