目前没有整理的

（1）文档为英文：

Cluster (集群)

Crypto (加密)

DNS (域名服务器)

Domain (域)

Process (进程)

punycode

TLS (安全传输层)

V8

VM (虚拟机)

ZLIB (压缩)

1. 暂时无用

C/C++ 插件

Debugger (调试器)

1 Assert (断言)

断好失 等深严两否 扔不 如

assert(value[, message])

assert.ok(value[, message])

assert.fail(actual, expected, message, operator)

assert.equal(actual, expected[, message]) assert.notEqual(actual, expected[, message])

assert.deepEqual(actual, expected[, message]) assert.notDeepEqual(actual, expected[, message])

assert.strictEqual(actual, expected[, message]) assert.notStrictEqual(actual, expected[, message])

assert.deepStrictEqual(actual, expected[, message]) assert.notDeepStrictEqual(actual, expected[, message])

assert.throws(block[, error][, message])

assert.doesNotThrow(block[, error][, message])

assert.ifError(value)

2 Buffer

缓池从4分不慢 是缓编 字比接

长项 换三

读写有无大小 正 16 32

读写有无 8

读写大小 浮达

转串j前后 比复入等填包键段值写

类

Buffer.poolSize

Buffer.from(array)

Buffer.from(arrayBuffer[, byteOffset[, length]])

Buffer.from(buffer)

Buffer.from(string[, encoding])

Buffer.alloc(size[, fill[, encoding]])

Buffer.allocUnsafe(size)

Buffer.allocUnsafeSlow(size)

Buffer.isBuffer(obj)

Buffer.isEncoding(encoding)

Buffer.byteLength(string[, encoding])

Buffer.compare(buf1, buf2)

Buffer.concat(list[, totalLength])

实例

buf.length

buf[index]

buf.swap16()

buf.swap32()

buf.swap64()

buf.readIntBE(offset, byteLength[, noAssert])

buf.readInt16BE(offset[, noAssert])

buf.readInt32BE(offset[, noAssert])

buf.readIntLE(offset, byteLength[, noAssert])

buf.readInt16LE(offset[, noAssert])

buf.readInt32LE(offset[, noAssert])

buf.readUInt16BE(offset[, noAssert])

buf.readUInt32BE(offset[, noAssert])

buf.readUIntBE(offset, byteLength[, noAssert])

buf.readUInt16LE(offset[, noAssert])

buf.readUInt32LE(offset[, noAssert])

buf.readUIntLE(offset, byteLength[, noAssert])

buf.writeIntBE(value, offset, byteLength[, noAssert])

buf.writeInt16BE(value, offset[, noAssert])

buf.writeInt32BE(value, offset[, noAssert])

buf.writeIntLE(value, offset, byteLength[, noAssert])

buf.writeInt16LE(value, offset[, noAssert])

buf.writeInt32LE(value, offset[, noAssert])

buf.writeUIntBE(value, offset, byteLength[, noAssert])

buf.writeUInt16BE(value, offset[, noAssert])

buf.writeUInt32BE(value, offset[, noAssert])

buf.writeUIntLE(value, offset, byteLength[, noAssert])

buf.writeUInt16LE(value, offset[, noAssert])

buf.writeUInt32LE(value, offset[, noAssert])

buf.readInt8(offset[, noAssert])

buf.readUInt8(offset[, noAssert])

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

buf.writeUInt8(value, offset[, noAssert])

buf.readFloatBE(offset[, noAssert])

buf.readDoubleBE(offset[, noAssert])

buf.readFloatLE(offset[, noAssert])

buf.readDoubleLE(offset[, noAssert])

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

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

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

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

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

buf.toJSON()

buf.indexOf(value[, byteOffset][, encoding])

buf.lastIndexOf(value[, byteOffset][, encoding])

buf.compare(target[, targetStart[, targetEnd[, sourceStart[,

sourceEnd]]]])

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

buf.entries()

buf.equals(otherBuffer)

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

buf.includes(value[, byteOffset][, encoding])

buf.keys()

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

buf.values()

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

3 Child Processes (子进程)

子执文产异叉

消断关错退

标连管进出错

发断杀

child\_process.exec(command[, options][, callback])

child\_process.execFile(file[, args][, options][, callback])

child\_process.spawn(command[, args][, options])

child\_process.execFileSync(file[, args][, options])

child\_process.execSync(command[, options])

child\_process.spawnSync(command[, args][, options])

child\_process.fork(modulePath[, args][, options])

'message' 事件

'disconnect' 事件

'close' 事件

'error' 事件

'exit' 事件

child.pid

child.connected

child.stdio

child.stdin

child.stdout

child.stderr

child.send(message[, sendHandle[, options]][, callback])

child.disconnect()

child.kill([signal])

4 CLI (命令行选项)

命版帮评印查互要 不赞警 追赞警同 扔零保轨专安能迫开vi

排路废替终额i

-v, --version

-h, --help

-e, --eval "script"

-p, --print "script"

-c, --check

-i, --interactive

-r, --require module

--no-deprecation

--no-warnings

--trace-deprecation

--trace-warnings

--trace-sync-io

--throw-deprecation

--zero-fill-buffers

--preserve-symlinks

--track-heap-objects

--prof-process

--tls-cipher-list=list

--enable-fips

--force-fips

--openssl-config=file

--v8-options

--icu-data-dir=file

环境变量

NODE\_DEBUG=module[,…]

NODE\_PATH=path[:…]

NODE\_DISABLE\_COLORS=1

NODE\_REPL\_HISTORY=file

NODE\_TTY\_UNSAFE\_ASYNC=1

NODE\_EXTRA\_CA\_CERTS=file

NODE\_ICU\_DATA=file

5 Console (控制台)

控新时尾断目错信日追警

new Console(stdout[, stderr])

console.time(label)

console.timeEnd(label)

console.assert(value[, message][, ...args])

console.dir(obj[, options])

console.error([data][, ...args])

console.info([data][, ...args])

console.log([data][, ...args])

console.trace(message[, ...args])

console.warn([data][, ...args])

6 Error (错误)

错新捕堆 信堆

范引语类

Error 类

new Error(message)

Error.captureStackTrace(targetObject[, constructorOpt])

Error.stackTraceLimit

error.message

error.stack

RangeError 类

ReferenceError 类

SyntaxError 类

TypeError 类

7 Events (事件)

事新移 默

设得 听数 加移所 预一 监一 触事

'newListener' 事件

'removeListener' 事件

EventEmitter.defaultMaxListeners

emitter.setMaxListeners(n)

emitter.getMaxListeners()

emitter.listeners(eventName)

emitter.listenerCount(eventName)

emitter.addListener(eventName, listener)

emitter.removeListener(eventName, listener)

emitter.removeAllListeners([eventName])

emitter.prependListener(eventName, listener)

emitter.prependOnceListener(eventName, listener)

emitter.on(eventName, listener)

emitter.once(eventName, listener)

emitter.emit(eventName[, ...args])

emitter.eventNames()

8 File System (文件系统)

文观变错 关

状

读写 开关字路

异同读写 读文

异同空lf 改用访状

异同空f 缩戳

异同 通添开关存真重移写 f数同 连不符 建目临 读目连

常 创读写 看文不

fs.FSWatcher 类

'change' 事件

'error' 事件

watcher.close()

fs.Stats 类

Stat 时间值

fs.ReadStream 类

'open' 事件

'close' 事件

readStream.bytesRead

readStream.path

fs.WriteStream 类

'open' 事件

'close' 事件

writeStream.bytesWritten

writeStream.path

fs.read(fd, buffer, offset, length, position, callback)

fs.readFile(file[, options], callback)

fs.write(fd, buffer, offset, length[, position], callback)

fs.writeFile(file, data[, options], callback)

fs.readSync(fd, buffer, offset, length, position)

fs.readFileSync(file[, options])

fs.writeSync(fd, buffer, offset, length[, position])

fs.writeFileSync(file, data[, options])

fs.chown(path, uid, gid, callback)

fs.chmod(path, mode, callback)

fs.stat(path, callback)

fs.lchown(path, uid, gid, callback)

fs.lchmod(path, mode, callback)

fs.lstat(path, callback)

fs.fchown(fd, uid, gid, callback)

fs.fchmod(fd, mode, callback)

fs.fstat(fd, callback)

fs.chownSync(path, uid, gid)

fs.chmodSync(path, mode)

fs.statSync(path)

fs.lchownSync(path, uid, gid)

fs.lchmodSync(path, mode)

fs.lstatSync(path)

fs.fchownSync(fd, uid, gid)

fs.fchmodSync(fd, mode)

fs.fstatSync(fd)

fs.truncate(path, len, callback)

fs.utimes(path, atime, mtime, callback)

fs.ftruncate(fd, len, callback)

fs.futimes(fd, atime, mtime, callback)

fs.truncateSync(path, len)

fs.utimesSync(path, atime, mtime)

fs.ftruncateSync(fd, len)

fs.futimesSync(fd, atime, mtime)

fs.access(path[, mode], callback)

fs.appendFile(file, data[, options], callback)

fs.open(path, flags[, mode], callback)

fs.close(fd, callback)

fs.exists(path, callback)

fs.realpath(path[, options], callback)

fs.rename(oldPath, newPath, callback)

fs.rmdir(path, callback)

fs.write(fd, data[, position[, encoding]], callback)

fs.fdatasync(fd, callback)

fs.fsync(fd, callback)

fs.link(existingPath, newPath, callback)

fs.unlink(path, callback)

fs.symlink(target, path[, type], callback)

fs.mkdir(path[, mode], callback)

fs.mkdtemp(prefix[, options], callback)

fs.readdir(path[, options], callback)

fs.readlink(path[, options], callback)

fs.accessSync(path[, mode])

fs.appendFileSync(file, data[, options])

fs.openSync(path, flags[, mode])

fs.closeSync(fd)

fs.existsSync(path)

fs.realpathSync(path[, options])

fs.renameSync(oldPath, newPath)

fs.rmdirSync(path)

fs.writeSync(fd, data[, position[, encoding]])

fs.fdatasyncSync(fd)

fs.fsyncSync(fd)

fs.linkSync(existingPath, newPath)

fs.unlinkSync(path)

fs.symlinkSync(target, path[, type])

fs.mkdirSync(path[, mode])

fs.mkdtempSync(prefix[, options])

fs.readdirSync(path[, options])

fs.readlinkSync(path[, options])

fs.constants

fs.createReadStream(path[, options])

fs.createWriteStream(path[, options])

fs.watch(filename[, options][, listener])

fs.watchFile(filename[, options], listener)

fs.unwatchFile(filename[, listener])

9 Global (全局变量)

全目文 设清立间尽 缓控全模进出 要藏解

\_\_dirname

\_\_filename

setImmediate(callback[, ...args])

setInterval(callback, delay[, ...args])

setTimeout(callback, delay[, ...args])

clearImmediate(immediateObject)

clearInterval(intervalObject)

clearTimeout(timeoutObject)

Buffer 类

console

global

module

process

exports

require()

require.cache

require.resolve()

10 HTTP

传代新 套大免请 创毁得

客 连升 完终 继回套 完终尾刷写 设不套尽

服 连名升 检继期 客关请 大听时 听3尽关

响 关完 完头发 状码信 写继头 得设移头 添尾尽

进 终关 版方套地头追 生头追 状码信 毁尽

方状全 请得创

http.Agent 类

new Agent([options])

agent.sockets

agent.maxSockets

agent.freeSockets

agent.maxFreeSockets

agent.requests

agent.createConnection(options[, callback])

agent.destroy()

agent.getName(options)

http.ClientRequest 类

'connect' 事件

'upgrade' 事件

'aborted' 事件

'abort' 事件

'continue' 事件

'response' 事件

'socket' 事件

request.aborted

request.abort()

request.end([data][, encoding][, callback])

request.flushHeaders()

request.write(chunk[, encoding][, callback])

request.setNoDelay([noDelay])

request.setSocketKeepAlive([enable][, initialDelay])

request.setTimeout(timeout[, callback])

http.Server 类

'connect' 事件

'connection' 事件

'upgrade' 事件

'checkContinue' 事件

'checkExpectation' 事件

'clientError' 事件

'close' 事件

'request' 事件

server.maxHeadersCount

server.listening

server.timeout

server.listen(handle[, callback])

server.listen(path[, callback])

server.listen([port][, hostname][, backlog][, callback])

server.setTimeout(msecs, callback)

server.close([callback])

http.ServerResponse 类

'close' 事件

'finish' 事件

response.finished

response.headersSent

response.sendDate

response.statusCode

response.statusMessage

response.write(chunk[, encoding][, callback])

response.writeContinue()

response.writeHead(statusCode[, statusMessage][, headers])

response.getHeader(name)

response.setHeader(name, value)

response.removeHeader(name)

response.addTrailers(headers)

response.end([data][, encoding][, callback])

response.setTimeout(msecs, callback)

http.IncomingMessage 类

'aborted' 事件

'close' 事件

message.httpVersion

message.method

message.socket

message.url

message.headers

message.trailers

message.rawHeaders

message.rawTrailers

message.statusCode

message.statusMessage

message.destroy([error])

message.setTimeout(msecs, callback)

http.METHODS

http.STATUS\_CODES

http.globalAgent

http.request(options[, callback])

http.get(options[, callback])

http.createServer([requestListener])

11 HTTPS

安代

服时尽

创听3关

全请得

https.Agent 类

https.Server 类

server.timeout

server.setTimeout(msecs, callback)

https.createServer(options[, requestListener])

server.listen(handle[, callback])

server.listen(path[, callback])

server.listen(port[, host][, backlog][, callback])

server.close([callback])

https.globalAgent

https.request(options, callback)

https.get(options, callback)

12 Module (模块)

模快出子文身载父要

exports 快捷方式

module.exports

module.children

module.filename

module.id

module.loaded

module.parent

module.require(id)

13 Net (网络)

网服 连错关听

听大 地关得 听4 涉不

套新 连错关数排尾查时

字读写 当地端 远地端家 缓连毁

连3 设编保不尽 涉不 地毁尾暂恢写

创服连3 连3 是空四六

net.Server 类

'connection' 事件

'error' 事件

'close' 事件

'listening' 事件

server.listening

server.maxConnections

server.address()

server.close([callback])

server.getConnections(callback)

server.listen(handle[, backlog][, callback])

server.listen(options[, callback])

server.listen(path[, backlog][, callback])

server.listen([port][, hostname][, backlog][, callback])

server.ref()

server.unref()

net.Socket 类

new net.Socket([options])

'connect' 事件

'error' 事件

'close' 事件

'data' 事件

'drain' 事件

'end' 事件

'lookup' 事件

'timeout' 事件

socket.bytesRead

socket.bytesWritten

socket.localAddress

socket.localPort

socket.remoteAddress

socket.remotePort

socket.remoteFamily

socket.bufferSize

socket.connecting

socket.destroyed

socket.connect(options[, connectListener])

socket.connect(path[, connectListener])

socket.connect(port[, host][, connectListener])

socket.setEncoding([encoding])

socket.setKeepAlive([enable][, initialDelay])

socket.setNoDelay([noDelay])

socket.setTimeout(timeout[, callback])

socket.ref()

socket.unref()

socket.address()

socket.destroy([exception])

socket.end([data][, encoding])

socket.pause()

socket.resume()

socket.write(data[, encoding][, callback])

net.createServer([options][, connectionListener])

net.createConnection(options[, connectListener])

net.createConnection(path[, connectListener])

net.createConnection(port[, host][, connectListener])

net.connect(options[, connectListener])

net.connect(path[, connectListener])

net.connect(port[, host][, connectListener])

net.isIP(input)

net.isIPv4(input)

net.isIPv6(input)

14 OS (操作系统)

操末常

核架顺空家主负网平释临总类上用

os.EOL

os.constants

os.cpus()

os.arch()

os.endianness()

os.freemem()

os.homedir()

os.hostname()

os.loadavg()

os.networkInterfaces()

os.platform()

os.release()

os.tmpdir()

os.totalmem()

os.type()

os.uptime()

os.userInfo([options])

15 Path (路径)

路win po 片路 基目扩格是加正析相解

path.win32

path.posix

path.sep

path.delimiter

path.basename(path[, ext])

path.dirname(path)

path.extname(path)

path.format(pathObject)

path.isAbsolute(path)

path.join([...paths])

path.normalize(path)

path.parse(path)

path.relative(from, to)

path.resolve([...paths])

16 Query Strings (查询字符串)

查逃不 析串

querystring.escape(str)

querystring.unescape(str)

querystring.parse(str[, sep[, eq[, options]]])

querystring.stringify(obj[, sep[, eq[, options]]])

17 Readline (逐行读取)

逐接 关暂恢线 亢cz恢 关暂恢提设问写

清行屏 创完光移发

Interface 类

'close' 事件

'pause' 事件

'resume' 事件

'line' 事件

'SIGINT' 事件

'SIGTSTP' 事件

'SIGCONT' 事件

rl.close()

rl.pause()

rl.resume()

rl.prompt([preserveCursor])

rl.setPrompt(prompt)

rl.question(query, callback)

rl.write(data[, key])

readline.clearLine(stream, dir)

readline.clearScreenDown(stream)

readline.createInterface(options)

completer 函数的使用

readline.cursorTo(stream, x, y)

readline.moveCursor(stream, dx, dy)

readline.emitKeypressEvents(stream[, interface])

18 REPL (交互式解释器)

交服退重定展 开

REPLServer 类

'exit' 事件

'reset' 事件

replServer.defineCommand(keyword, cmd)

replServer.displayPrompt([preserveCursor])

repl.start([options])

19 Stream (流)

流写关错 管不 排完

塞不 尾设写

关错 数尾读

是暂管非 读恢设不包

两改

stream.Writable 类

'close' 事件

'error' 事件

'pipe' 事件

'unpipe' 事件

'drain' 事件

'finish' 事件

writable.cork()

writable.uncork()

writable.end([chunk][, encoding][, callback])

writable.setDefaultEncoding(encoding)

writable.write(chunk[, encoding][, callback])

stream.Readable 类

'close' 事件

'error' 事件

'data' 事件

'end' 事件

'readable' 事件

readable.isPaused()

readable.pause()

readable.pipe(destination[, options])

readable.unpipe([destination])

readable.read([size])

readable.resume()

readable.setEncoding(encoding)

readable.unshift(chunk)

readable.wrap(stream)

stream.Duplex 类

stream.Transform 类

20 String Decoder (字符串解码器)

字新尾写

类: new StringDecoder([encoding])

stringDecoder.end([buffer])

stringDecoder.write(buffer)

21 Timer (定时器)

定立

时涉非

设清立间尽

Immediate 类

Timeout 类

timeout.ref()

timeout.unref()

预定定时器

setImmediate(callback[, ...args])

setInterval(callback, delay[, ...args])

setTimeout(callback, delay[, ...args])

取消定时器

clearImmediate(immediate)

clearInterval(timeout)

clearTimeout(timeout)

22 TTY (终端)

终读是设

写重 列行

是

tty.ReadStream 类

readStream.isRaw

readStream.setRawMode(mode)

tty.WriteStream 类

'resize' 事件

writeStream.columns

writeStream.rows

tty.isatty(fd)

23 UDP/Datagram (数据报)

数套关错听信

设4绑2涉不 添地关掉发

模创2

dgram.Socket 类

'close' 事件

'error' 事件

'listening' 事件

'message' 事件

socket.setBroadcast(flag)

socket.setMulticastLoopback(flag)

socket.setMulticastTTL(ttl)

socket.setTTL(ttl)

socket.bind([port][, address][, callback])

socket.bind(options[, callback])

socket.ref()

socket.unref()

socket.addMembership(multicastAddress[, multicastInterface])

socket.address()

socket.close([callback])

socket.dropMembership(multicastAddress[, multicastInterface])

socket.send(msg, [offset, length,] port, address[, callback])

dgram module functions

dgram.createSocket(options[, callback])

dgram.createSocket(type[, callback])

24 URL

统哈主名完路端协查 砍权后问

格析解

URL 字符串与 URL 对象

urlObject.hash

urlObject.host

urlObject.hostname

urlObject.href

urlObject.pathname

urlObject.port

urlObject.protocol

urlObject.search

urlObject.slashes

urlObject.auth

urlObject.path

urlObject.query

url.format(urlObject)

url.parse(urlString[, parseQueryString[, slashesDenoteHost]])

url.resolve(from, to)

25 Util (实用工具)

工排反格继检

util.debuglog(section)

util.deprecate(function, string)

util.format(format[, ...args])

util.inherits(constructor, superConstructor)

util.inspect(object[, options])

1 Assert (断言)

断好失 等深严两否 扔不 如

2 Buffer

缓池从4分不慢 是缓编 字比接

长项 换三

读写有无大小 正 16 32

读写有无 8

读写大小 浮达

转串j前后 比复入等填包键段值写

3 Child Processes (子进程)

子执文产异叉 消断关错退 标连管进出错 发断杀

4 CLI (命令行选项)

命版帮评印查互要 不赞警 追赞警同 扔零保轨专安能迫开vi 排路废替终额i

5 Console (控制台)

控新时尾断目错信日追警

6 Error (错误)

错新捕堆信堆 范引语类

7 Events (事件)

事新移默 设得听数加移所 预一监一触事

8 File System (文件系统)

文观变错 关状

读写 开关字路

异同读写 读文

异同空lf 改用访状

异同空f 缩戳

异同 通添开关存真重移写 f数同连不符 建目临读目连

常创读写看文不

9 Global (全局变量)

全目文 设清立间尽 缓控全模进出 要藏解

10 HTTP

传代新 套大免请 创毁得

客连升 完终继回套 完终尾刷写 设不套尽

服连名升 检继期 客关请 大听时 听3尽关

响关完 完头发 状码信 写继头 得设移头 添尾尽

进终关 版方套地头追 生头追 状码信 毁尽

方状全 请得创

11 HTTPS

安代服时尽 创听3关 全请得

12 Module (模块)

模快出子文身载父要

13 Net (网络)

网服 连错关听 听大地关得 听4涉不

套新 连错关数排尾查时

字读写当地端 远地端家缓连毁

连3 设编保不尽 涉不地毁尾暂恢写

创服连3 连3 是空四六

14 OS (操作系统)

操末常 核架顺空家主负网平释临总类上用

15 Path (路径)

路win po 片路 基目扩格是加正析相解

16 Query Strings (查询字符串)

查逃不 析串

17 Readline (逐行读取)

逐接 关暂恢线亢cz恢 关暂恢提设问写 清行屏创完光移发

18 REPL (交互式解释器)

交服退重定展开

19 Stream (流)

流写关错 管不排完

塞不尾设写 关错数尾读

是暂管非 读恢设不包

两改

20 String Decoder (字符串解码器)

字新尾写

21 Timer (定时器)

定立 时涉非 设清立间尽

22 TTY (终端)

终读是设 写重列行是

23 UDP/Datagram (数据报)

数套关错听信

设4绑2涉不 添地关掉发

模创2

24 URL

统哈主名完路端协查 砍权后问格析解

25 Util (实用工具)

工排反格继检

断缓子命控错事文 全传安模网操路查 逐交流字定终数统工

\* \* \* \* \* \*