String decoder

Source Code: lib/string_decoder.js

The `node:string_decoder` module provides an API for decoding `Buffer` objects into strings in a manner that preserves encoded multi-byte UTF-8 and UTF-16 characters. It can be accessed using:

```
const { StringDecoder } = require('node:string_decoder'); copy
```

The following example shows the basic use of the `StringDecoder` class.

```
const { StringDecoder } = require('node:string_decoder');
const decoder = new StringDecoder('utf8');

const cent = Buffer.from([0xC2, 0xA2]);
console.log(decoder.write(cent)); // Prints: ¢

const euro = Buffer.from([0xE2, 0x82, 0xAC]);
console.log(decoder.write(euro)); // Prints: € copy
```

When a `Buffer` instance is written to the `StringDecoder` instance, an internal buffer is used to ensure that the decoded string does not contain any incomplete multibyte characters. These are held in the buffer until the next call to `stringDecoder.write()` or until `stringDecoder.end()` is called.

In the following example, the three UTF-8 encoded bytes of the European Euro symbol (€) are written over three separate operations:

```
const { StringDecoder } = require('node:string_decoder');
const decoder = new StringDecoder('utf8');

decoder.write(Buffer.from([0xE2]));
decoder.write(Buffer.from([0x82]));
console.log(decoder.end(Buffer.from([0xAC]))); // Prints: € copy
```

Class: `StringDecoder`

`new StringDecoder([encoding])`

- encoding <string> The character encoding the `StringDecoder` will use. **Default:** `'utf8'`.

Added in: v0.1.99

Creates a new `StringDecoder` instance.

```
stringDecoder.end([buffer])
```

- buffer <string> | <Buffer> | <TypedArray> | <DataView> The bytes to decode.

- Returns: <string>

Returns any remaining input stored in the internal buffer as a string. Bytes representing incomplete UTF-8 and UTF-16 characters will be replaced with substitution characters appropriate for the character encoding.

If the buffer argument is provided, one final call to stringDecoder.write() is performed before returning the remaining input. After end() is called, the stringDecoder object can be reused for new input.

```
stringDecoder.write(buffer)
```

- buffer <string> | <Buffer> | <TypedArray> | <DataView> The bytes to decode.

▶ History

- Returns: <string>

Returns a decoded string, ensuring that any incomplete multibyte characters at the end of the Buffer, or TypedArray, or DataView are omitted from the returned string and stored in an internal buffer for the next call to stringDecoder.write() or stringDecoder.end() .

Exported from DevDocs — https://devdocs.io