build failing

ansi-styles

> *ANSI escape codes for styling strings in the terminal*

You probably want the higher-level chalk module for styling your strings.

Install

```
$ npm install ansi-styles
```

Usage

```
const style = require('ansi-styles');

console.log(`${style.green.open}Hello world!${style.green.close}`);


// Color conversion between 16/256/truecolor
// NOTE: If conversion goes to 16 colors or 256 colors, the original color
//       may be degraded to fit that color palette. This means terminals
//       that do not support 16 million colors will best-match the
//       original color.
console.log(style.bgColor.ansi.hsl(120, 80, 72) + 'Hello world!' +
style.bgColor.close);
console.log(style.color.ansi256.rgb(199, 20, 250) + 'Hello world!' +
style.color.close);
console.log(style.color.ansi16m.hex('#abcdef') + 'Hello world!' +
style.color.close);
```

API

Each style has an `open` and `close` property.

Styles

Modifiers

- `reset`
- `bold`
- `dim`
- `italic` *(Not widely supported)*
- `underline`
- `inverse`

- `hidden`
- `strikethrough` *(Not widely supported)*

Colors

- `black`
- `red`
- `green`
- `yellow`
- `blue`
- `magenta`
- `cyan`
- `white`
- `blackBright` (alias: `gray`, `grey`)
- `redBright`
- `greenBright`
- `yellowBright`
- `blueBright`
- `magentaBright`
- `cyanBright`
- `whiteBright`

Background colors

- `bgBlack`
- `bgRed`
- `bgGreen`
- `bgYellow`
- `bgBlue`
- `bgMagenta`
- `bgCyan`
- `bgWhite`
- `bgBlackBright` (alias: `bgGray`, `bgGrey`)
- `bgRedBright`
- `bgGreenBright`
- `bgYellowBright`
- `bgBlueBright`
- `bgMagentaBright`
- `bgCyanBright`
- `bgWhiteBright`

Advanced usage

By default, you get a map of styles, but the styles are also available as groups. They are non-enumerable so they don't show up unless you access them explicitly. This makes it easier to expose only a subset in a higher-level module.

- `style.modifier`
- `style.color`
- `style.bgColor`

EXAMPLE

```
console.log(style.color.green.open);
```

Raw escape codes (i.e. without the CSI escape prefix `\u001B[` and render mode postfix `m`) are available under `style.codes`, which returns a `Map` with the open codes as keys and close codes as values.

EXAMPLE

```
console.log(style.codes.get(36));
//=> 39
```

256 / 16 million (TrueColor) support

`ansi-styles` uses the `color-convert` package to allow for converting between various colors and ANSI escapes, with support for 256 and 16 million colors.

The following color spaces from `color-convert` are supported:

- `rgb`
- `hex`
- `keyword`
- `hsl`
- `hsv`
- `hwb`
- `ansi`
- `ansi256`

To use these, call the associated conversion function with the intended output, for example:

```
style.color.ansi.rgb(100, 200, 15); // RGB to 16 color ansi foreground code
style.bgColor.ansi.rgb(100, 200, 15); // RGB to 16 color ansi background code

style.color.ansi256.hsl(120, 100, 60); // HSL to 256 color ansi foreground code
style.bgColor.ansi256.hsl(120, 100, 60); // HSL to 256 color ansi foreground code

style.color.ansi16m.hex('#C0FFEE'); // Hex (RGB) to 16 million color foreground code
style.bgColor.ansi16m.hex('#C0FFEE'); // Hex (RGB) to 16 million color background
code
```

Related

- ansi-escapes - ANSI escape codes for manipulating the terminal

Maintainers

- Sindre Sorhus
- Josh Junon

For enterprise

Available as part of the Tidelift Subscription.

The maintainers of `ansi-styles` and thousands of other packages are working with Tidelift to deliver commercial support and maintenance for the open source dependencies you use to build your applications. Save time, reduce risk, and improve code health, while paying the maintainers of the exact dependencies you use. Learn more.