# How to

# print colorful text?

**Author: Nick Lee**

| 二進制 | 十六進制 | 十進制 |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 10 | 2 | 2 |
| 11 | 3 | 3 |
| 100 | 4 | 4 |
| 101 | 5 | 5 |
| 110 | 6 | 6 |
| 111 | 7 | 7 |
| 1000 | 8 | 8 |
| 1001 | 9 | 9 |
| 1010 | A | 10 |
| 1011 | B | 11 |
| 1100 | C | 12 |
| 1101 | D | 13 |
| 1110 | E | 14 |
| 1111 | F | 15 |

**2 7 4**

十進制

$4 \times 10^0 = 4$

$7 \times 10^1 = 70$

$2 \times 10^2 = 200$

$274$

**1 1 0 1**

二進制

$1 \times 2^0 = 1$

$0 \times 2^1 = 0$

$1 \times 2^2 = 4$

$1 \times 2^3 = 8$

$13$

**1 B**

十六進制

$11 \times 16^0 = 11$

$1 \times 16^1 = 16$

$27$

# 8 Colors

```
>>> print('\x1b[31m')
>>> print('abc')
```

| | |
|---|---|
| Black | \x1b[30m |
| Red | \x1b[31m |
| Green | \x1b[32m |
| Yellow | \x1b[33m |
| Blue | \x1b[34m |
| Magenta | \x1b[35m |
| Cyan | \x1b[36m |
| White | \x1b[37m |
| Reset | \x1b[0m |

經常打 `\x1b[..m` 好麻煩，容易一點好嗎？

可以編寫一個 `module`，將麻煩嘢收藏在一個個 `function` 之內……

# ansi.py

```
def code(a):
    return '\x1b[' + str(a) + 'm'

def red(s):
    return code(31) + s + code(0)

def green(s):
    return code(32) + s + code(0)

def yellow(s):
    return code(33) + s + code(0)
```

```
>>> import ansi
>>> print(ansi.red('abc'))
>>> print(ansi.red('red'), ansi.green('green'))
```

改過 module，記住 reload：

```
>>> from importlib import reload
>>> reload(ansi)
```

# 16 Colors

| | |
|---|---|
| Bright Black | \x1b[30;1m |
| Bright Red | \x1b[31;1m |
| Bright Green | \x1b[32;1m |
| Bright Yellow | \x1b[33;1m |
| Bright Blue | \x1b[34;1m |
| Bright Magenta | \x1b[35;1m |
| Bright Cyan | \x1b[36;1m |
| Bright White | \x1b[37;1m |
| Reset | \x1b[0m |

# 256 Colors

\x1b[38;5;${ID}m

顏色多了，原本的 module 應付不來，應如何修改？

# ansi.py
# (version 2)

```
def code(*args):  ❸
    return '\x1b[' + ';'.join([str(i) for i in args]) + 'm'

def red(s, bright=False):  ❶
    if bright:
        return code(31, 1) + s + code(0)
    else:
        return code(31) + s + code(0)

def green(s, bright=False):
    if bright:
        return code(32, 1) + s + code(0)
    else:
        return code(32) + s + code(0)

def yellow(s, bright=False):
    if bright:
        return code(33, 1) + s + code(0)
    else:
        return code(33) + s + code(0)

def color(id, s):  ❷
    return code(38, 5, id) + s + code(0)
```

❶ 給它一個控制色彩亮度、有預設值的 argument

❷ 作多一個 function，供 256 種顏色之用

❸ code()需要應付一、二、或三個參數

很多樣子差不多的段落，如何簡化？

# ansi.py
# (version 3)

```
def code(*args):
    return '\x1b[' + ';'.join([str(i) for i in args]) + 'm'

def _color16(s, base, bright):  ❶
    if bright:
        return code(base, 1) + s + code(0)
    else:
        return code(base) + s + code(0)

def red(s, bright=False):
    return _color16(s, 31, bright)

def green(s, bright=False):
    return _color16(s, 32, bright)

def yellow(s, bright=False):
    return _color16(s, 33, bright)

def color(id, s):
    return code(38, 5, id) + s + code(0)
```

❶「私家」function 通常以 underscore 開頭作標示

# list_colors.py

```
import ansi

for i in range(0, 256):
    print(ansi.color(i, str(i)), end=' ')
```

印得靚啲，例如一個 16x16 的方陣，得唔得？

# list_colors.py
# (version 2)

```
import ansi

for i in range(0, 16):
    for j in range(0, 16):
        id = i * 16 + j
        print(ansi.color(id, str(id)), end=' ')
    print()
```

仲係對唔齊，怎辦？

# list_colors.py
# (version 3)

```
import ansi

for i in range(0, 16):
    for j in range(0, 16):
        id = i * 16 + j
        print(ansi.color(id, str(id).rjust(3)), end=' ')
    print()
```