This document describes the escape sequences interpreted by tmux, the terminal multiplexer. These are the codes received by tmux from its controlled pty, not the codes sent from tmux clients to their controlling ptys.

This is intended to be a fairly complete technical reference for all the codes tmux understands or processes. It is based on reading the source code from version 1.8 of tmux.

For those who just care about text formatting, jump to table 6.

Most of these codes are handled in `input.c`

Table 1: Symbols in this document

Symbol	Meaning
ESC	The ASCII escape character. 033, 0x1b, 27
BEL	The ASCII Bell character. 007, 0x07, 7
SPC	ASCII Space. 040, 0x20, 32
{text}	Arbitrary(ish) text string. Most characters accepted.
{cmd}	A single command character, typically a letter.
{N}	A numeric parameter.

Table 2: Legacy Sequences

Sequence	Meaning
ESC (0	G0 Special Graphics Mode
ESC 7	Save Cursor
ESC 8	Restore Cursor
ESC # 8	Alignment test (fill with 'E')
ESC =	Keypad Application Mode
ESC >	Keypad Numeric Mode (default)
ESC (B	G0 ASCII Mode
ESC D	Index (move cursor down, scroll if at bottom)
ESC E	New Line (move cursor down and to home column)
ESC H	Set Horizontal Tab Stop
ESC M	Reverse Index (move cursor up)
ESC c	Reset to Initial State

Table 3: APC/DSC/Etc codes

Sequence	Meaning
ESC _ {text} ESC \	ASC: Set pty (pane) title to {text}
ESC] {text} BEL	OSC: See table 4 for format of {text}
ESC [{text} {cmd}	CSI: See table 5 for alphabetically sorted list of commands
ESC P {text} ESC \	DSC: ignore {text}

ESC	X	{text}	ESC	\	SOS: ignored?TODO
ESC	^				PM ??TODO
ESC	k	{text}	ESC	\	rename_string??TODO

Table 4: OSC Codes

Sequence						Meaning
ESC	]	0	;	{text}	BEL	screen_set_title?TODO
ESC	]	2	;	{text}	BEL	screen_set_title?TODO
ESC	]	1	2	;	{text} BEL	set cursor color to {text}?TODO
ESC	]	1	1	2	; BEL	set cursor color to default?TODO

1 CSI Sequences

CSI starts with ESC [, has an optional private extension character (<,>,or ?), followed by an optional list of semicolon separated numbers of an arbitrary number of digits, and ends with any ASCII char between 0x40 and 0x7E, which is almost always a letter[1]

The numbers are refered to as 'parameters'. Different commands expect different numbers of parameters. Extra parameters are ignored. Missing parameters have a default value.

Some commands have positional arguments (e.g. put cursor at X,Y has 2 positional arguments, the first is always X, the second always Y). Some commands have accumulative mode arguments (e.g. SGR has bold, italic; it doesn't matter which order they are specified, though they are applied in the order given and some modes replace others. For example, specifying blue,green will set the color to blue and then green, resulting in green). Some are mixed (SGR is typically accumulative, except for codes 38 and 48, which are extended set parameter values which many terminals interpret as switching to positional parameter interpretation for the number of expected parameters, then return to accumulative interpretation. See section 1.1 for how tmux handles this).

Below is a table of CSI commands that tmux understands and acts upon.

Table 5: CSI Codes

Code				sym	Meaning
ESC	[	{N}	@	ICH	Insert char N (TODO?)TODO
ESC	[	{N}	A	CUU	cursor up N times, min 1 def 1
ESC	[	{N}	B	CUD	cursor down N times, min 1 def 1

[1]Non letter symbols include @, [,], \, ^, _, `, {, }, ~, and |. Of these, tmux only takes action on @.

ESC [{N} C	CUF	cursor right N times, min 1 def 1
ESC [{N} D	CUB	cursor left N times, min 1 def 1
ESC [{N} E	CNL	caraige return, cursor down N times, min 1 def 1
ESC [{N} F	CPL	caraige ret, cursor up N times, min 1 def 1
ESC [{N} G	HPA	set curor to column N (min 1 def 1)
ESC [{N} ; {M} H	CUP	cursor to N x M (min 1 def 1 for both) (N is row, M is column)
(no I)		
ESC [{N} ; {M} J	ED	clear based on N (min 0 def 0) N = 0: clear to end of screen N = 1: clear to top of screen N = 2: clear whole screen N = 3: clear based on M M = 0: clear history (LINUX console, used for locking screen) M = other: ignored N = other: ignored
ESC [{N} K	EL	Clear based on N (min 0 def 0) N = 0: Clear to end of line N = 1: Clear to start of line N = 2: Clear whole line N = other: ignored
ESC [{N} L	IL	Insert line N (TODO?)TODO
ESC [{N} M	DL	Delete Line based on N (TODO)TODO
(no N)		
(no O)		
ESC [{N} P	DCH	Delete based on N (TODO)TODO
(no Q)		
(no R)		
(no S)		
(no T)		
(no U)		
(no V)		
(no W)		
ESC [{N} X	ECH	clear based on N (TODO)TODO
(no Y)		

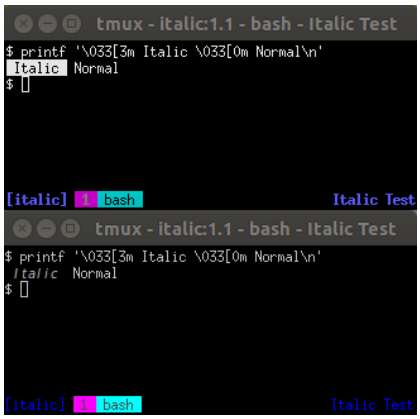
ESC [{N} Z	CBT	cursor back tab N times, min 1, default 1
(no a)		
(no b)		
<ESC> [0 c	DA	reply <ESC> [? 1 ; 2 c
<ESC> [other c		ignoreTODO
<ESC> [> 0 c	DA_TWO	reply <ESC> [> 0 ; 9 5 ; 0 c
<ESC> [> other c		ignoreTODO
ESC [d	VPA	
(no e)		
ESC [f	CUP	
ESC [g	TBC	
ESC [{N} h	SM	n min 0 def -1 N = 4: (IRM) Mode set (INSERT) N = other: ignored
ESC [? {N} h	SM_PRIVATE	n min 0 def -1 N = 1: (GATM) Mode set KCURSOR N = 3: (DECCOLM) cursor move to 0,0; clear scrn N = 7: (DECAWM) mode set wrap N = 25: (TCEM) mode set cursor N = 1000: mode clear ALL_MOUSE, mode set MOUSE STANDARD N = 1002: clear ALL_MOUSE, set MOUSE_BUTTON N = 1003: clear ALL_MOUSE, set MOUSE_ANY N = 1004: FOCUSON? (TODO)TODO[OTHERS: TODO]TODO
(no i)		
(no j)		
(no k)		
ESC [{N} l	RM	n min 0 def -1 N = 4: (IRM) Mode clear (MODE_INSERT) N = other: ignored

ESC [? {N} l	RM_PRIVATE	n min 0 def -1 N = 1: (GATM) Mode clear (MODE_KCURSOR) N = 3: (DECCOLM) cursor move to 0,0; clear screen N = 7: (DECAWM) Mode clear (MODE_WRAP) N = 25: (TCEM) Mode clear (MODE_CURSOR) N = 1000-1003: Mode clear (ALL_MOUSE_MODES) N = 1004: Mode clear (FOCUSON) N = 1005: Mode clear (Mouse UTF8) N = 1006: Mode clear MOUSE_SGR N = 47 or 1047: alternate pane off 0 (TODO)TODO N = 1049: alternate pane off 1 (TODO)TODO N = 2004: Mode clear (BRACKETPASTE) N = other: ignored
ESC [{text} m	SGR	see section 1.1
ESC [{N} n	DSR	if N is 5, reply <ESC> [0 n if N is 6, reply <ESC> [ROW ; COL R other N ignored
(no o)		
(no p)		
ESC [SPC <?> q	DECSCUSR	TODO
ESC [{N} ; {M} r	DECSTBM	scroll region N to M (M defaults to screen height)
ESC [s	SCP	Save cursor position
(no t)		
ESC [u	RCP	Restore cursor position
(no v-z)		

1.1 SGR Codes (color and other attributes)

The SGR codes can be given in any order but apply in the order given. tmux supports up to 16 codes specified in a single CSI sequence. Each code is separated by a semicolon as is normal convention.

An example: ESC [3 ; 3 7 m would set italic, foreground color 7, and would leave other settings alone (e.g. if it was bold, it is still bold, the



Top terminal is xterm (XTerm(297))
bottom terminal is urxvt (rxvt-unicode (urxvt) v9.19 - released: 2013-10-27)

Figure 1: Comparison of display of italic attribute.

background color is whatever it was, etc.)

Note: As with other features interpreted by tmux, the actual display seen by the user will depend on the capabilities of the terminal emulator which is running the tmux client. For example, a terminal which doesn't understand italics will not show italic text, even though tmux knows it should be italic. For several of these (italics being a good example), tmux will use the terminfo for your terminal emulator and do what that says, which can result in italic text being displayed as reverse text, for example.

Table 6: SGR Codes

Code	Meaning
0	Reset all to default (except alt-charset, see section 2)
1	set ATTR_Bright (e.g. "bold")
2	set ATTR_Dim
3	set ATTR_Italics
4	set ATTR_Underscore (i.e. "underline")
5	set ATTR_Blink
6	(ignored)
7	set ATTR_Reverse
8	set ATTR_Hidden
9	(ignored)
10	Same as 0.
11–21	(ignored)
22	Unset Bright and unset Dim
23	Unset Italic
24	Unset Underscore
25	Unset Blink

26	(ignored)
27	Unset Reverse
28–29	(ignored)
30–37	Set foreground to N-30, non-256color mode
38	foreground 256 color spec. See section 1.2
39	Set foreground color to 8, non-256color mode (terminal default)
40–47	Set background to N-40, non-256color mode
48	background 256 color spec. See section 1.2
49	Set background color to 8, non-256color mode
50–89	(ignored)
90–97	Set foreground color to N, non-256color mode (bright ANSI color)
98–99	(ignored)
100–107	Set background color to N-10, non-256color mode (bright ANSI color)
(others)	(ignored)

1.2 256 Color specification

tmux handles 256 color specifications of the form `5 ; {color}`. If a value other than 5 is received first, it is ignored and the next parameter is processed as a regular SGR value.

If the value after the 5 is missing (indicating default value), the color is chosen as color 8 in non-256 color mode (i.e. terminal default color).

Otherwise, `{color}` is a value between 0 and 255.

Colors from 0 to 7 correspond to non-256 colors 0 to 7 (black, red, green, yellow, blue, magenta, cyan, white).

Colors from 8 to 15 correspond to high-intensity non-256 colors 0 to 7 (either 90-97, or possibly 0 to 7 with BRIGHT attribute, depending on client terminal)

Colors 16 to 231 are colorwheel colors. Each of red, green, and blue colors get a number from 0 to 5 (0 being no intensity, 5 being full intensity). To go from color to code, use $16 + (r * 36) + (g * 6) + b$. To extract the color value from the code, use $r = \lfloor \frac{c}{36} \rfloor$, $g = \lfloor \frac{c \bmod 36}{6} \rfloor$, and $b = c \bmod 6$

2 Internal representation

This section contains some information about how tmux keeps track of things internally, which may be helpful in understanding how SGR codes affect things.

Every character on screen (and in history) has an 8-bit attribute bitmask, an 8-bit flag bitmask, an 8-bit foreground color, an 8-bit background color, 8-bit state (for tracking unicode info), and then unicode-width character data (9 bytes).

The attributes mask uses all 8 bit positions, and stores Bright, Dim, Underscore, Blink, Reverse, Hidden, Italics and Alt-Charset attributes.

Only 3 of the 8 flags are defined: Foreground is 256-color mode, Background is 256-color mode, and Padding.

Note that although Alt-Charset is stored with the attributes, it is not managed by SCI SGR. SGR 0 resets only the attributes managed by SGR.