NAME

Term::Cap - Perl termcap interface

SYNOPSIS

```
require Term::Cap;
$terminal = Tgetent Term::Cap { TERM => undef, OSPEED => $ospeed };
$terminal->Trequire(qw/ce ku kd/);
$terminal->Tgoto('cm', $col, $row, $FH);
$terminal->Tputs('dl', $count, $FH);
$terminal->Tpad($string, $count, $FH);
```

DESCRIPTION

These are low-level functions to extract and use capabilities from a terminal capability (termcap) database.

More information on the terminal capabilities will be found in the termcap manpage on most Unix-like systems.

METHODS

The output strings for **Tputs** are cached for counts of 1 for performance. **Tgoto** and **Tpad** do not cache. `$self->{_xx}` is the raw termcap data and `$self->{xx}` is the cached version.

```
print $terminal->Tpad($self->{_xx}, 1);
```

Tgoto, **Tputs**, and **Tpad** return the string and will also output the string to $FH if specified.

Tgetent

Returns a blessed object reference which the user can then use to send the control strings to the terminal using **Tputs** and **Tgoto**.

The function extracts the entry of the specified terminal type *TERM* (defaults to the environment variable *TERM*) from the database.

It will look in the environment for a *TERMCAP* variable. If found, and the value does not begin with a slash, and the terminal type name is the same as the environment string *TERM*, the *TERMCAP* string is used instead of reading a termcap file. If it does begin with a slash, the string is used as a path name of the termcap file to search. If *TERMCAP* does not begin with a slash and name is different from *TERM*, **Tgetent** searches the files *$HOME/.termcap*, */etc/termcap*, and */usr/share/misc/termcap*, in that order, unless the environment variable *TERMPATH* exists, in which case it specifies a list of file pathnames (separated by spaces or colons) to be searched **instead**. Whenever multiple files are searched and a tc field occurs in the requested entry, the entry it names must be found in the same file or one of the succeeding files. If there is a `:tc=...:` in the *TERMCAP* environment variable string it will continue the search in the files as above.

The extracted termcap entry is available in the object as `$self->{TERMCAP}`.

It takes a hash reference as an argument with two optional keys:

OSPEED

The terminal output bit rate (often mistakenly called the baud rate) for this terminal - if not set a warning will be generated and it will be defaulted to 9600. *OSPEED* can be specified as either a POSIX termios/SYSV termio speeds (where 9600 equals 9600) or an old DSD-style speed (where 13 equals 9600).

TERM

The terminal type whose termcap entry will be used - if not supplied it will default to $ENV{TERM}: if that is not set then **Tgetent** will croak.

It calls `croak` on failure.

Tpad

Outputs a literal string with appropriate padding for the current terminal.

It takes three arguments:

$string

The literal string to be output. If it starts with a number and an optional '*' then the padding will be increased by an amount relative to this number, if the '*' is present then this amount will be multiplied by $cnt. This part of $string is removed before output/

$cnt

Will be used to modify the padding applied to string as described above.

$FH

An optional filehandle (or IO::Handle) that output will be printed to.

The padded $string is returned.

Tputs

Output the string for the given capability padded as appropriate without any parameter substitution.

It takes three arguments:

$cap

The capability whose string is to be output.

$cnt

A count passed to Tpad to modify the padding applied to the output string. If $cnt is zero or one then the resulting string will be cached.

$FH

An optional filehandle (or IO::Handle) that output will be printed to.

The appropriate string for the capability will be returned.

Tgoto

Tgoto decodes a cursor addressing string with the given parameters.

There are four arguments:

$cap

The name of the capability to be output.

$col

The first value to be substituted in the output string (usually the column in a cursor addressing capability)

$row

The second value to be substituted in the output string (usually the row in cursor addressing capabilities)

$FH

An optional filehandle (or IO::Handle) to which the output string will be printed.

Substitutions are made with $col and $row in the output string with the following sprintf() line formats:

```
 %%   output `%'
```

```
%d   output value as in printf %d
%2   output value as in printf %2d
%3   output value as in printf %3d
%.   output value as in printf %c
%+x  add x to value, then do %.

%>xy if value > x then add y, no output
%r   reverse order of two parameters, no output
%i   increment by one, no output
%B   BCD (16*(value/10)) + (value%10), no output

%n   exclusive-or all parameters with 0140 (Datamedia 2500)
%D   Reverse coding (value - 2*(value%16)), no output (Delta Data)
```

The output string will be returned.

Trequire

Takes a list of capabilities as an argument and will croak if one is not found.

EXAMPLES

```
use Term::Cap;

# Get terminal output speed
require POSIX;
my $termios = new POSIX::Termios;
$termios->getattr;
my $ospeed = $termios->getospeed;

# Old-style ioctl code to get ospeed:
#     require 'ioctl.pl';
#     ioctl(TTY,$TIOCGETP,$sgtty);
#     ($ispeed,$ospeed) = unpack('cc',$sgtty);

# allocate and initialize a terminal structure
$terminal = Tgetent Term::Cap { TERM => undef, OSPEED => $ospeed };

# require certain capabilities to be available
$terminal->Trequire(qw/ce ku kd/);

# Output Routines, if $FH is undefined these just return the string

# Tgoto does the % expansion stuff with the given args
$terminal->Tgoto('cm', $col, $row, $FH);

# Tputs doesn't do any % expansion.
$terminal->Tputs('dl', $count = 1, $FH);
```

COPYRIGHT AND LICENSE

AUTHOR

This module is part of the core Perl distribution and is also maintained for CPAN by Jonathan Stowe <jns@gellyfish.co.uk>.

The code is hosted on Github: https://github.com/jonathanstowe/Term-Cap please feel free to fork, submit patches etc, etc there.

SEE ALSO

termcap(5)