The TTY, part 3

COMS10012 Software Tools

Colours

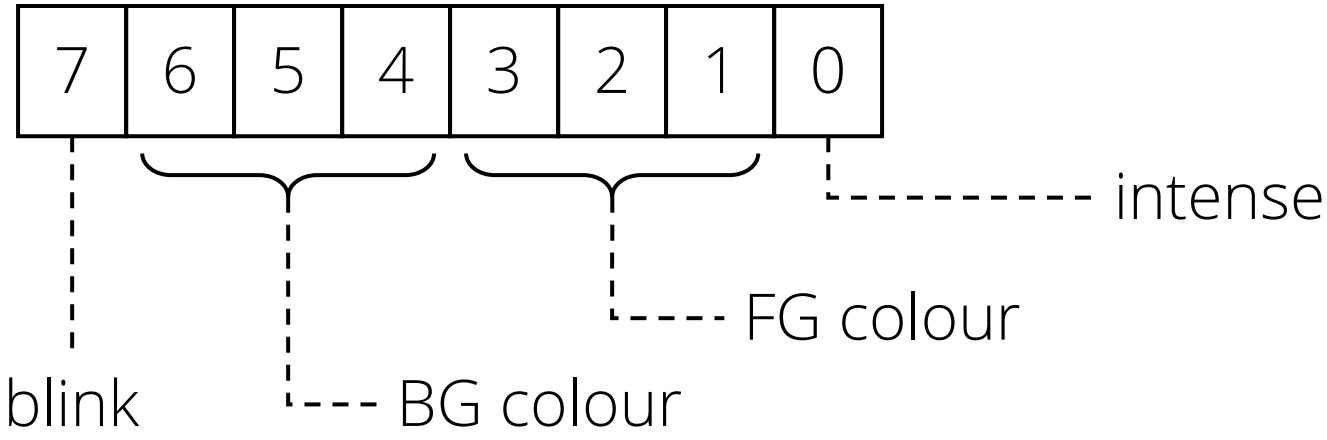
Terminal fonts

VGA (9x16 in original)

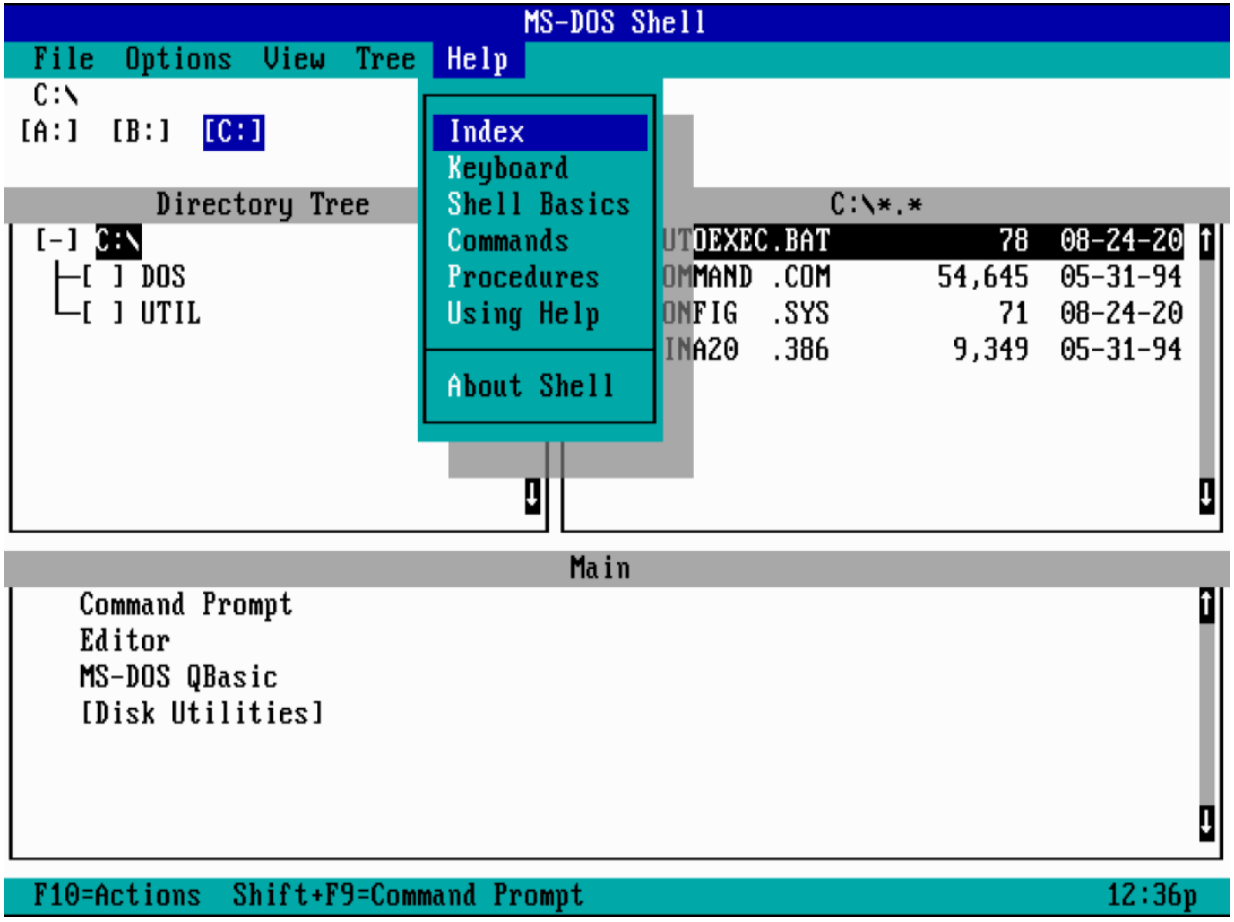
CGA, EGA, VGA ...

640x480 screen – 80x25 characters text mode

Video RAM: 2 bytes/character, 2nd is character, first one is mode:



dosshell


MS-DOS Shell
File Options View Tree Help
C:\
[A:] [B:] [C:]
Index
Keyboard
Shell Basics
Commands
Procedures
Using Help
About Shell
Directory Tree
C:*.*
[-] C:\
[ ] DOS
[ ] UTIL
UTOEXEC.BAT 78 08-24-20
OMMAND .COM 54,645 05-31-94
ONFIG .SYS 71 08-24-20
INA20 .386 9,349 05-31-94
Main
Command Prompt
Editor
MS-DOS QBasic
[Disk Utilities]
F10=Actions Shift+F9=Command Prompt 12:36p

colour codes

`ls` normally checks if its standard output is a tty before applying colours.

`ls --colour=always > ls.txt`

Trying to **`cat`** this file just shows coloured output. Open in in nano or similar to see what's going on.

```
csxdb@it075869:~/vagrant$ls
micro        sample-data.sql   user             Vagrantfile.mariadb
sampledata   secure-setup.sql  Vagrantfile      Vagrantfile.original
```

colour codes – escape sequences

```
^[[0m^[[38;5;34mmicro^[[0m
^[[38;5;27msampledata^[[0m
sample-data.sql
secure-setup.sql
...
```

```
csxdb@it075869:~/vagrant$ls
micro        sample-data.sql   user              Vagrantfile.mariadb
sampledata   secure-setup.sql  Vagrantfile       Vagrantfile.original
```

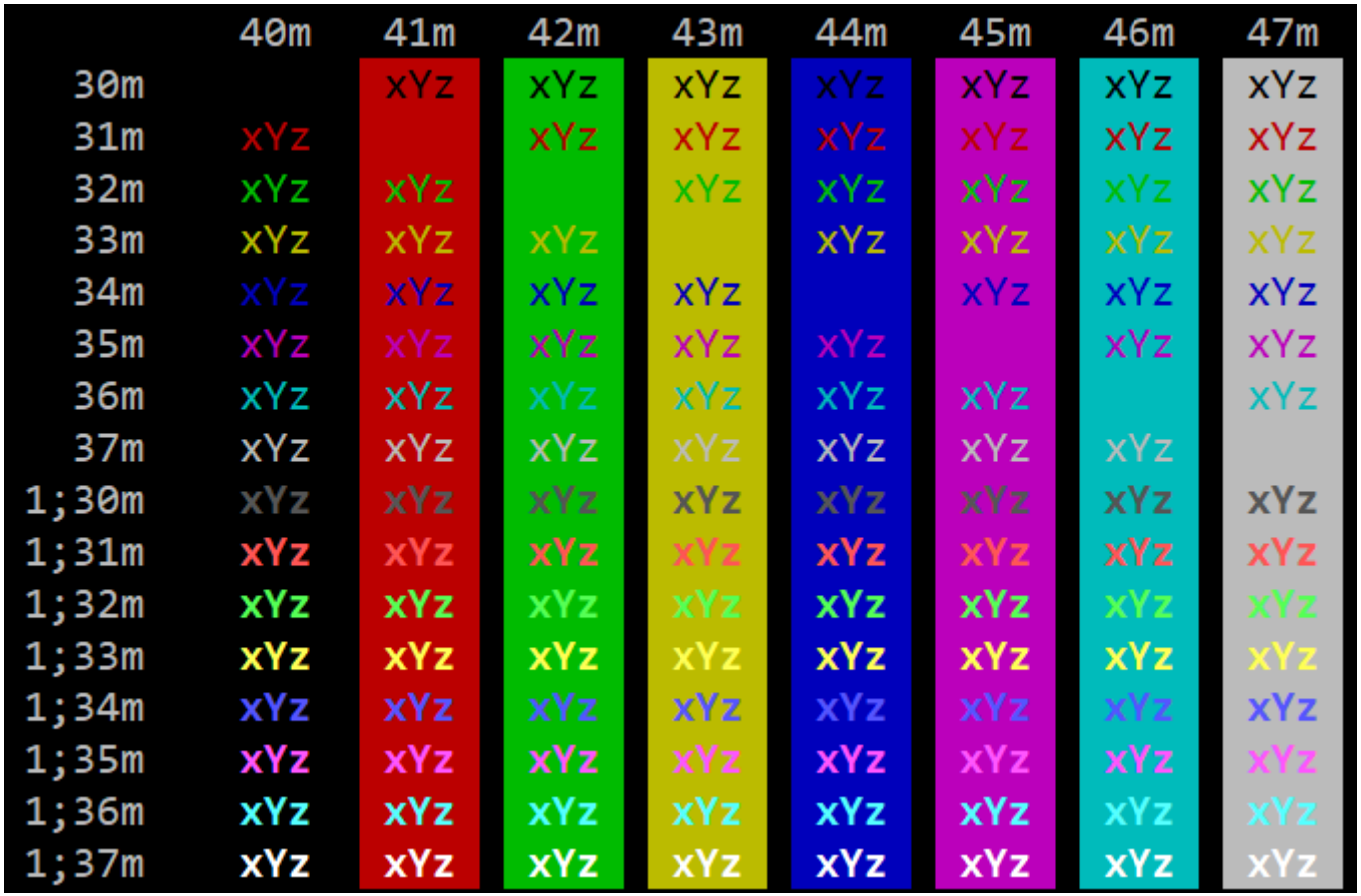
ANSI sequences

ESC (0x1B), opening square bracket
arguments (colour codes)
command (m = set colour)

e.g. **←[33;44m** = yellow on blue

```
$ echo -e '\e[34mBlue!\e[0m'
Blue!
```

colour codes



cursor movement

```
printf("%s", "0% done ...");
```

```
...
```

```
printf("%s", "\e[100D10% done ...");
```

D = move cursor left – note no newlines at end

note: **`setbuf(stdout, NULL);`** first!

Concurrency

Concurrency

In UNIX, something as simple as **`ls`** uses concurrency features: this is a two-process producer-consumer system.

Building this from scratch is hard – as you'll learn in "Computer Systems".

Luckily, the UNIX kernel can do most of the hard work for you.

Blocking IO

In UNIX, IO calls (read, write) normally block if there is nothing available on the "other end".

You can pass a parameter to request non-blocking IO, then you get an error if there's nothing to read or write.

Named pipes

```
$ mkfifo mypipe
$ ls -l
...
prw-r--r--.  1 user grp   4096 ... mypipe
```


Named pipes

```
$ mkfifo mypipe
$ yes > mypipe
(blocks)
```

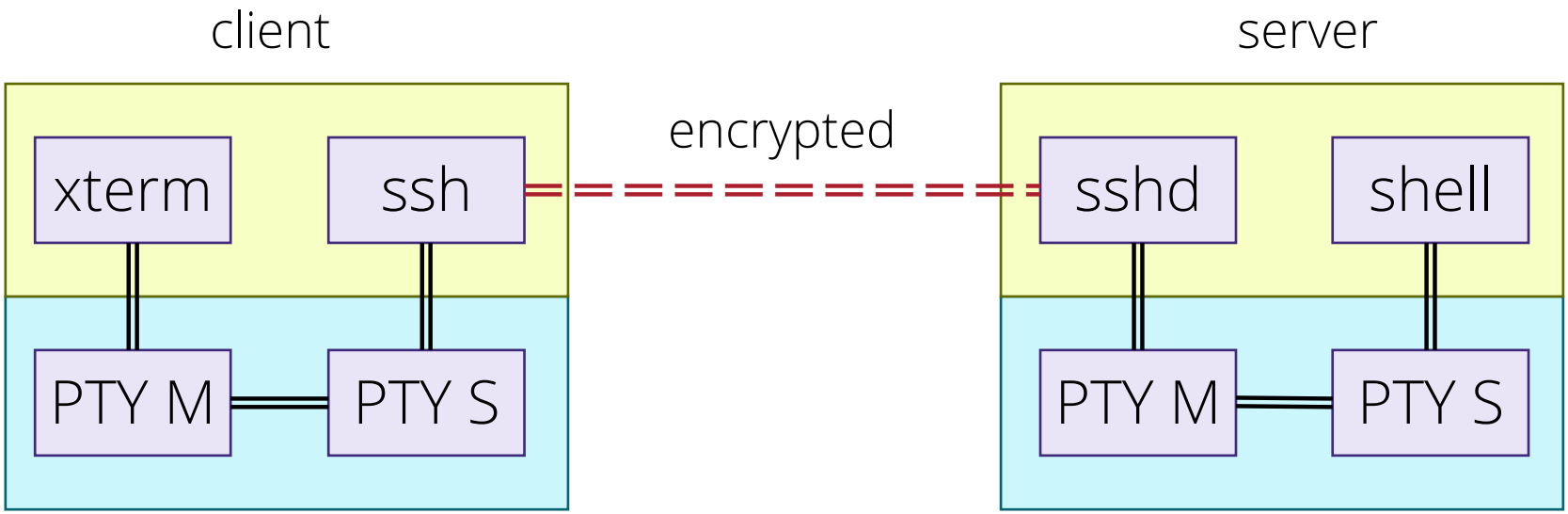
```
$
```

```
$ head -n 2 mypipe
y
y
$
```

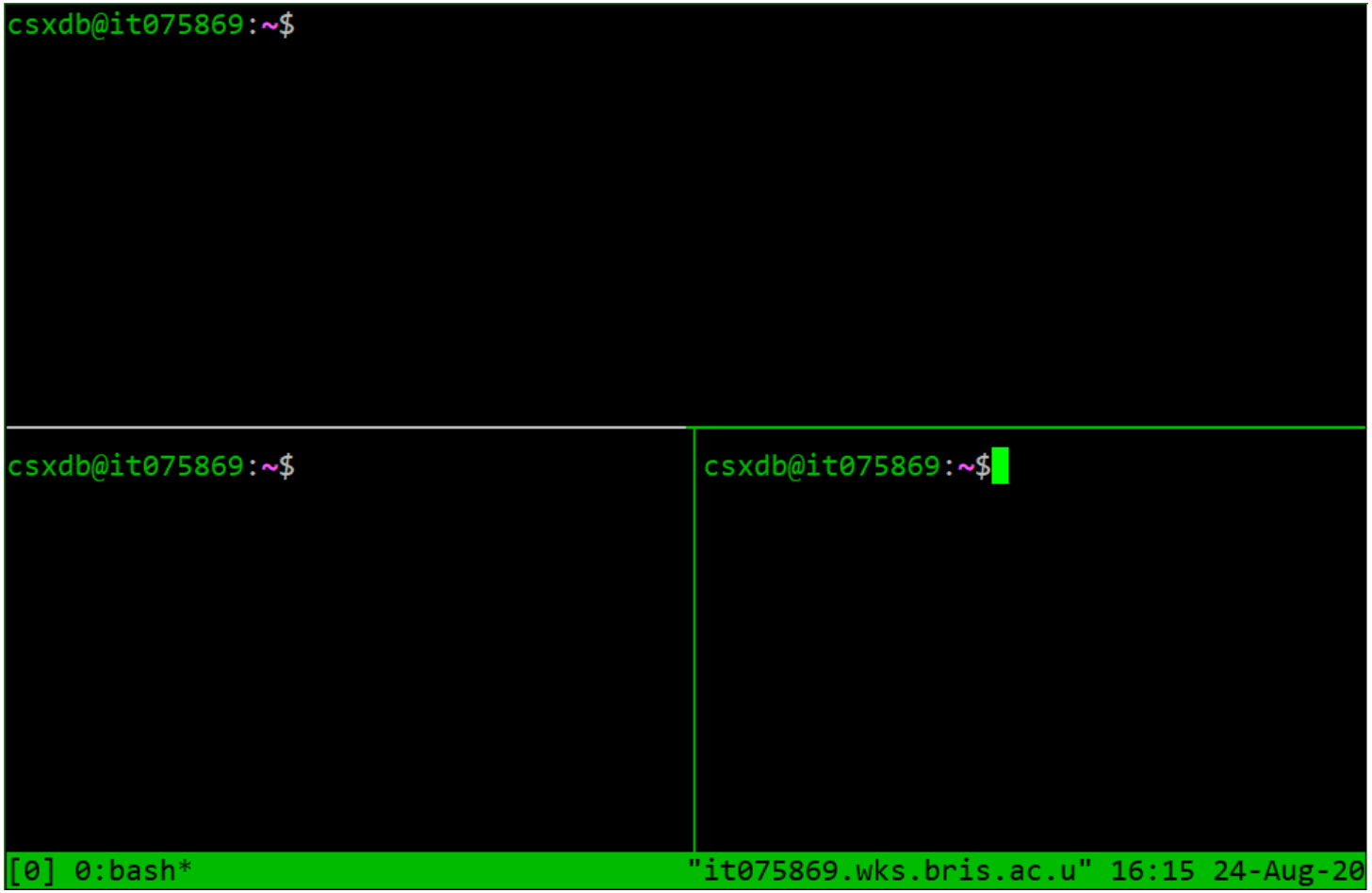

More terminals

ssh

SSH is a remote terminal:
both server and client open a PTY



terminals within terminals



tmux

`screen, tmux, byobu` : terminal multiplexers

tmux (on lab machines):
^B sends tmux command

- `^B "` = split vertical, `^B %` = split horizontal
- `^B` then arrows chooses terminal
- `^B c` creates a new window

tmux copy/paste

- `^B [` enters copy/scroll mode.
- client terminal is frozen, but you can scroll with arrows and Page Up/Down.
- To copy text, select start and **^SPACE**, move to end, then **^W**. To exit without copy, just **q**.
- `^B ]` pastes – works in other terminals in the same tmux session.

socat

`socat` is an extended **`cat`** that lets you connect to TCP sockets just like files. It even does TLS.

You can even connect a TCP socket to a PTY.

```
$ socat - TCP4:www.bristol.ac.uk:80
```

(replace "**`-`**" with **`readline`** if you want)

Interested in network security? Learn socat!

THE END