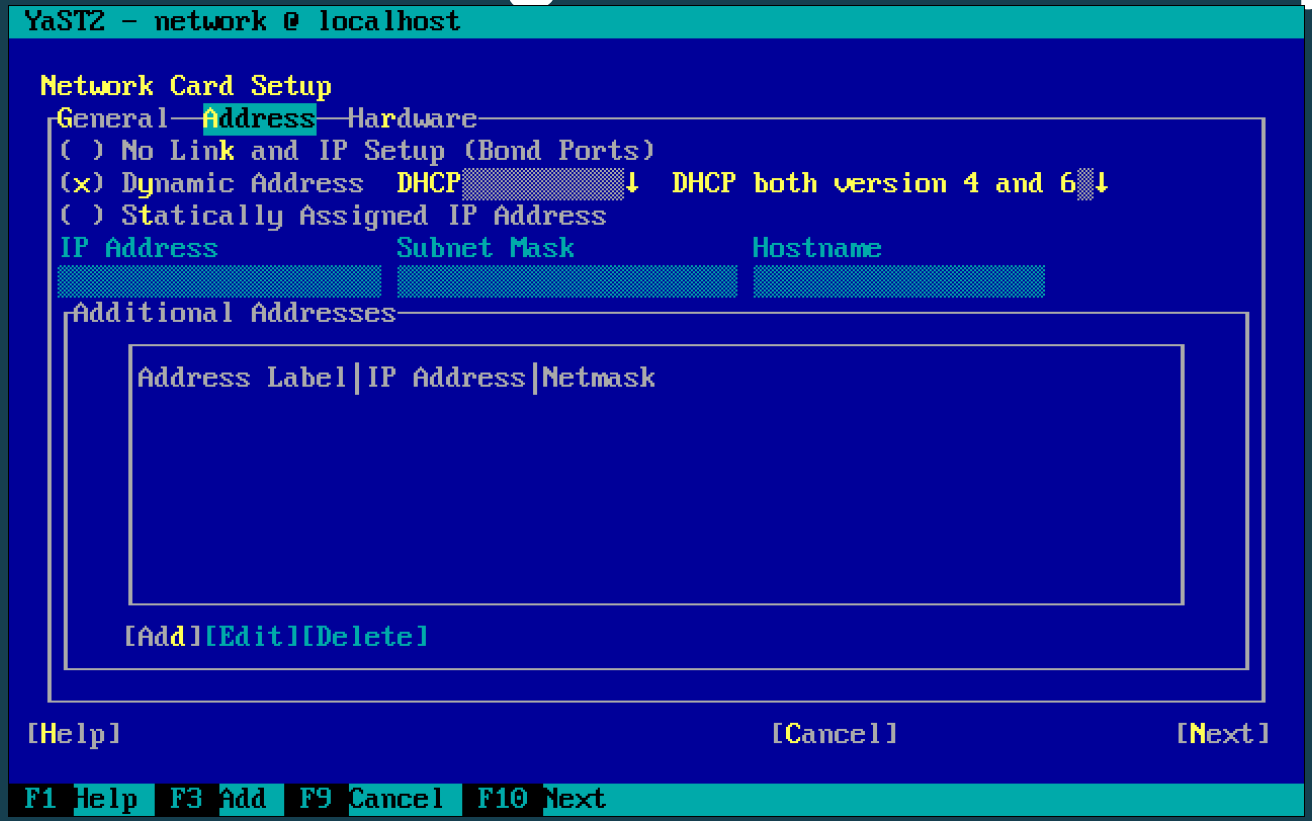
openSUSE Conference

# Accessibility of terminal color schemes

Jan Engelhardt <ej@inai.de>

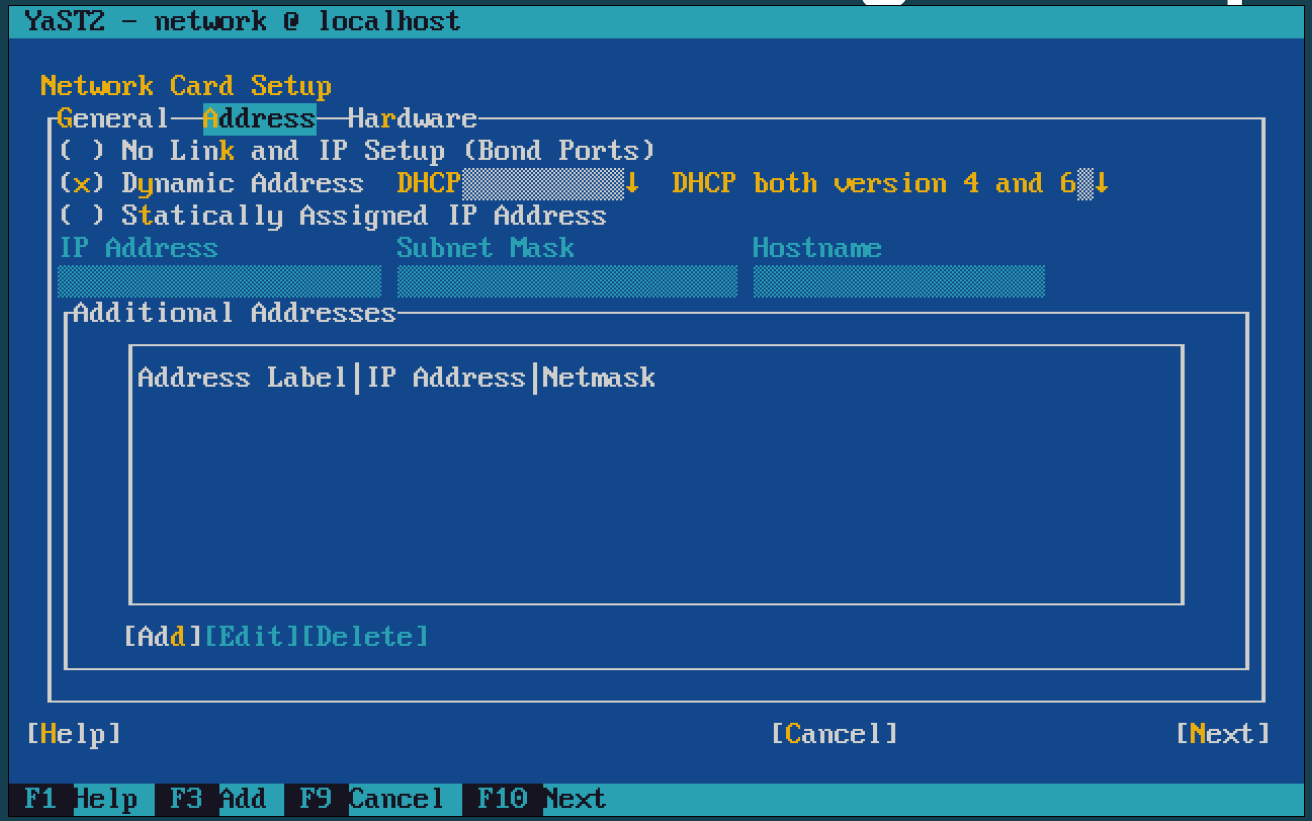
2024-06-29

# good ol' textmode yast2



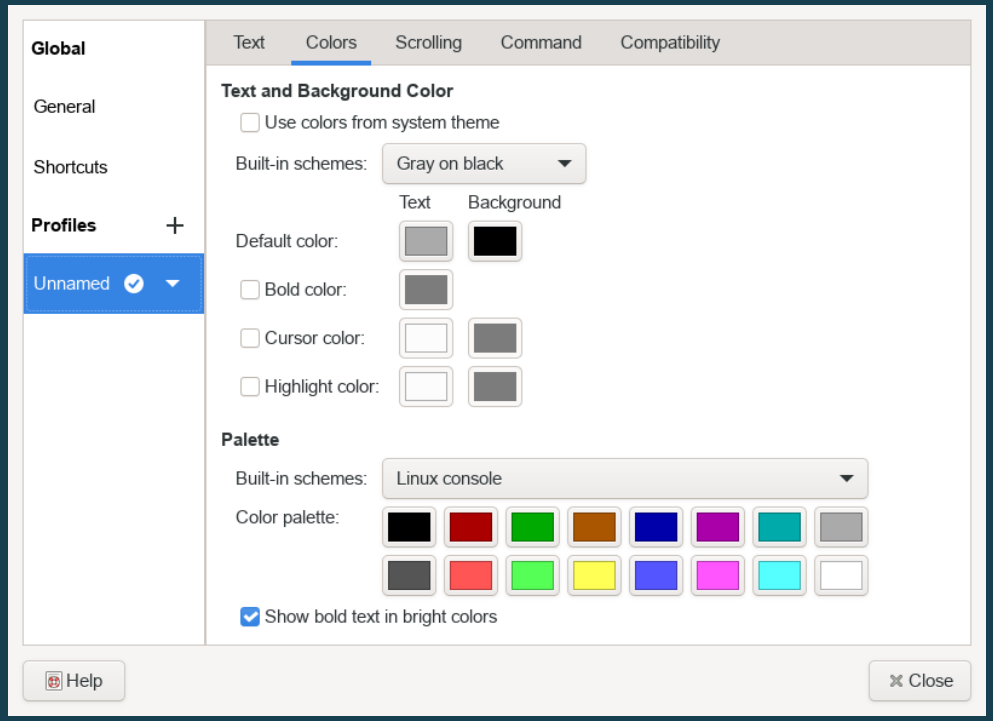

- You can do the same visual experiment with Midnight Commander

# …“gnome” palette



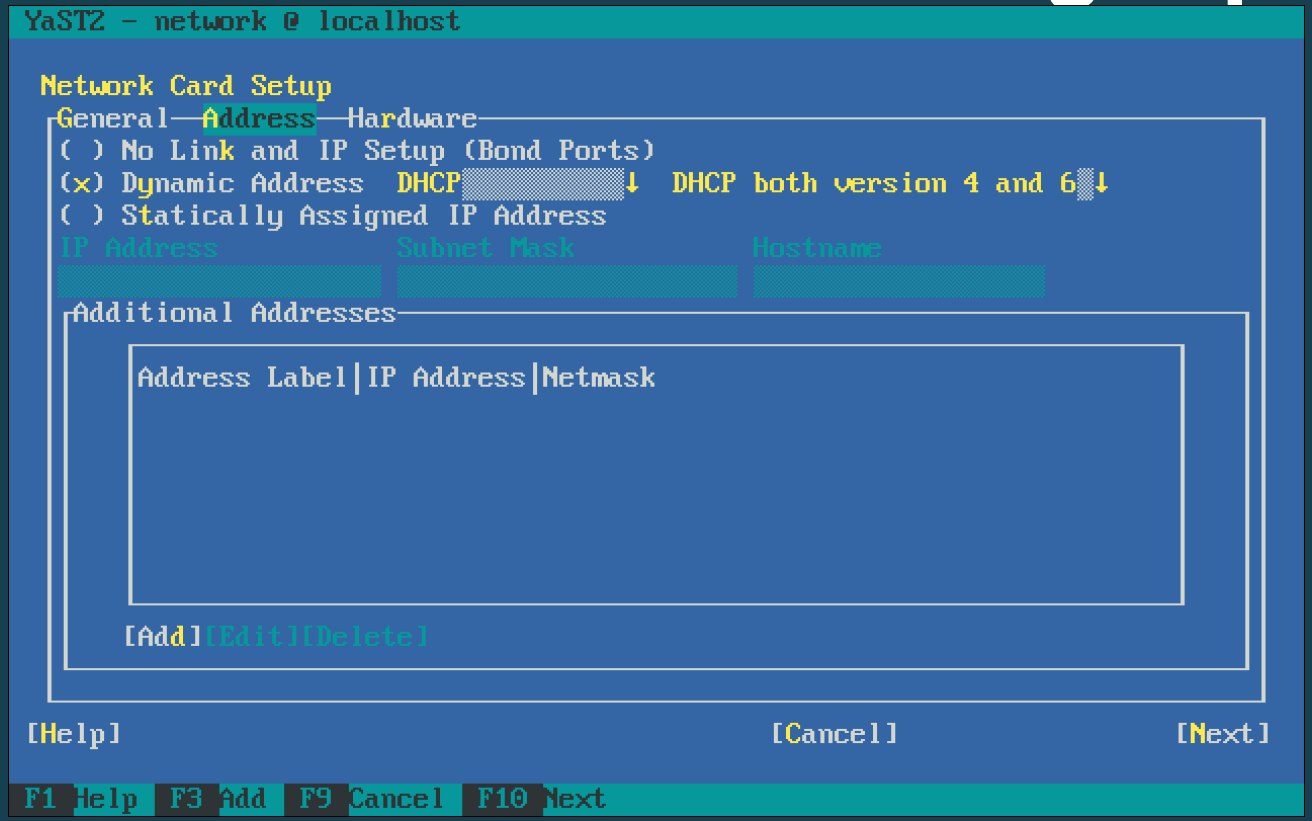

- Open yast2 in gnome-terminal and it looks like this

# ah yes, the choice

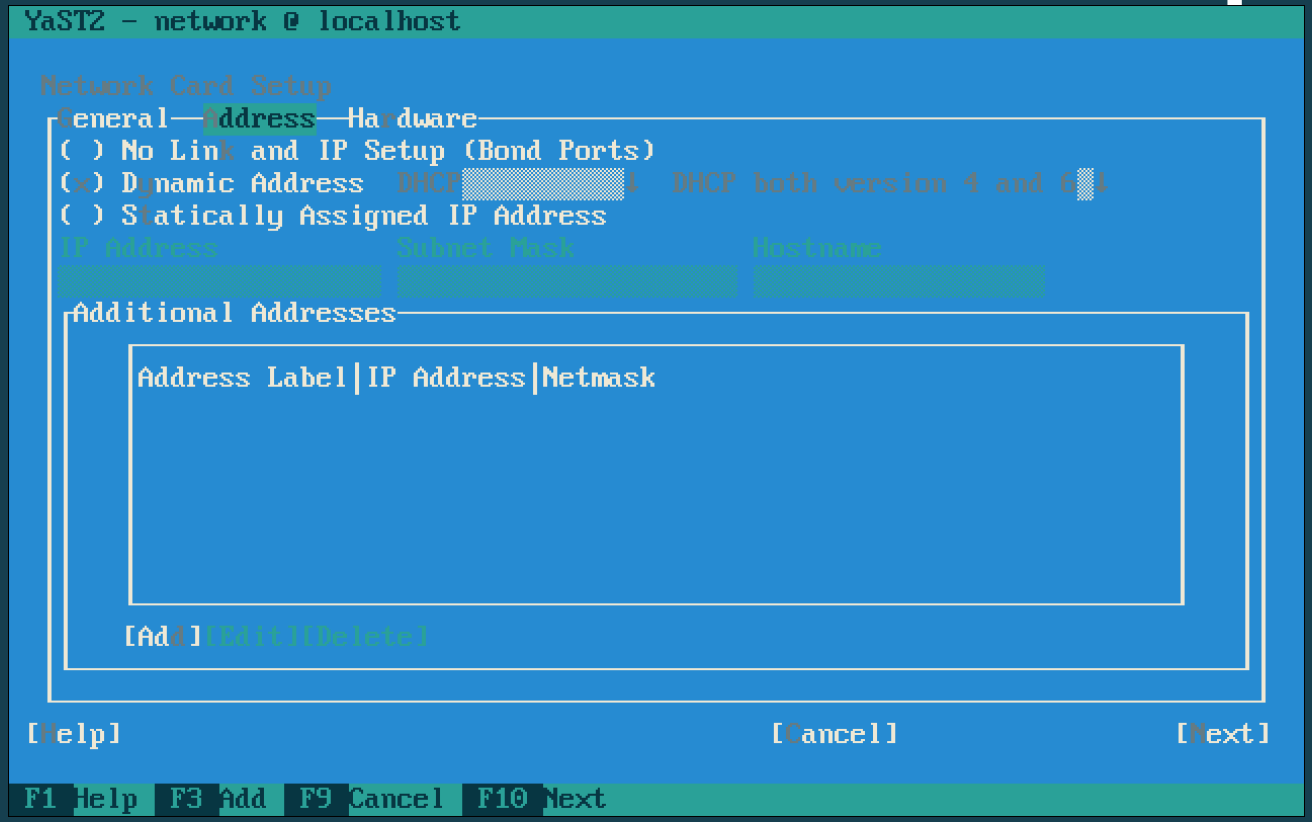



- terminal programs' defaults vary
- some offer a handful of built-in presets for personalization

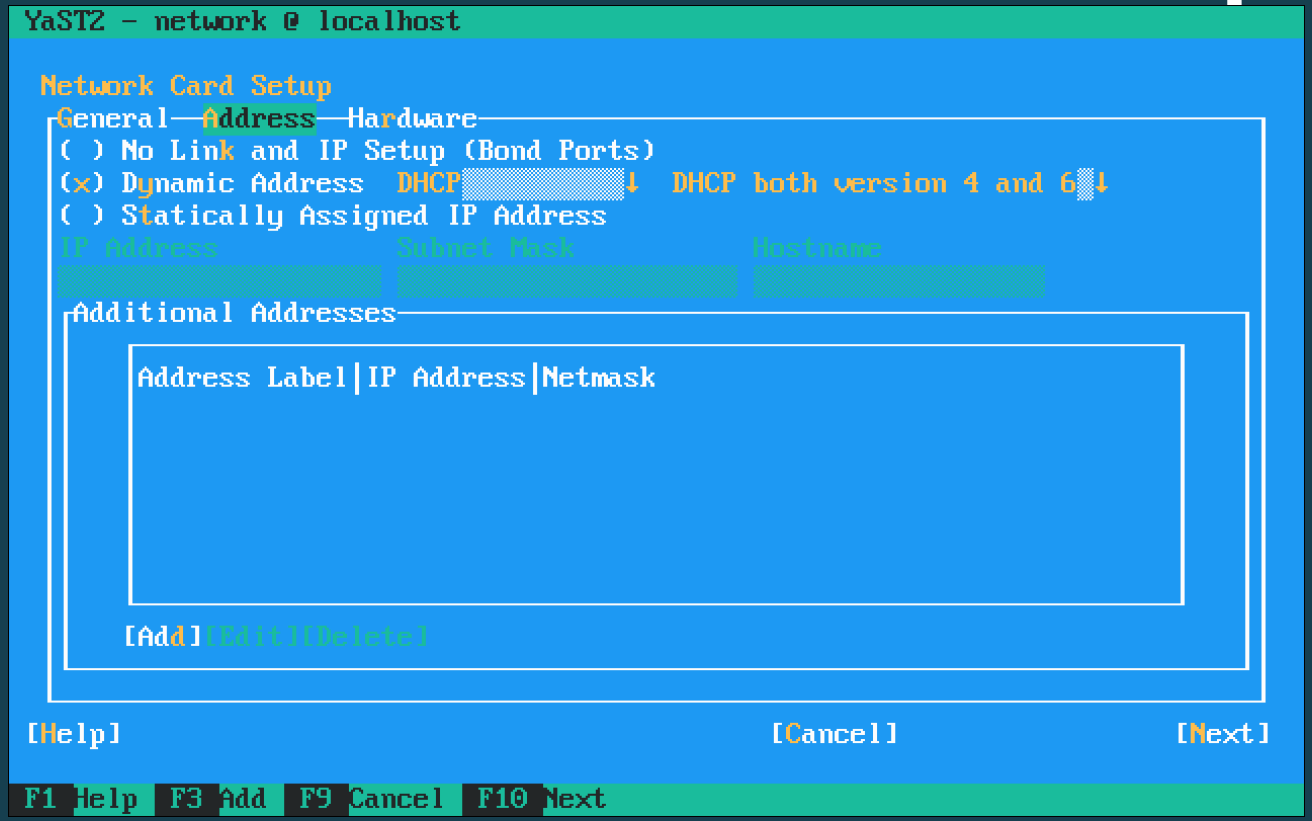
# …“tango” palette




# …“solarized” palette




# …“breeze” palette




- used by KDE Konsole

# Low contrast palettes

- bad accessibility
- want to quantify with numbers just how dire the situation is
- *I have just the tool for that…*
  `zypper in consoleet-utils` or
  https://codeberg.org/consoleet/consoleet-utils

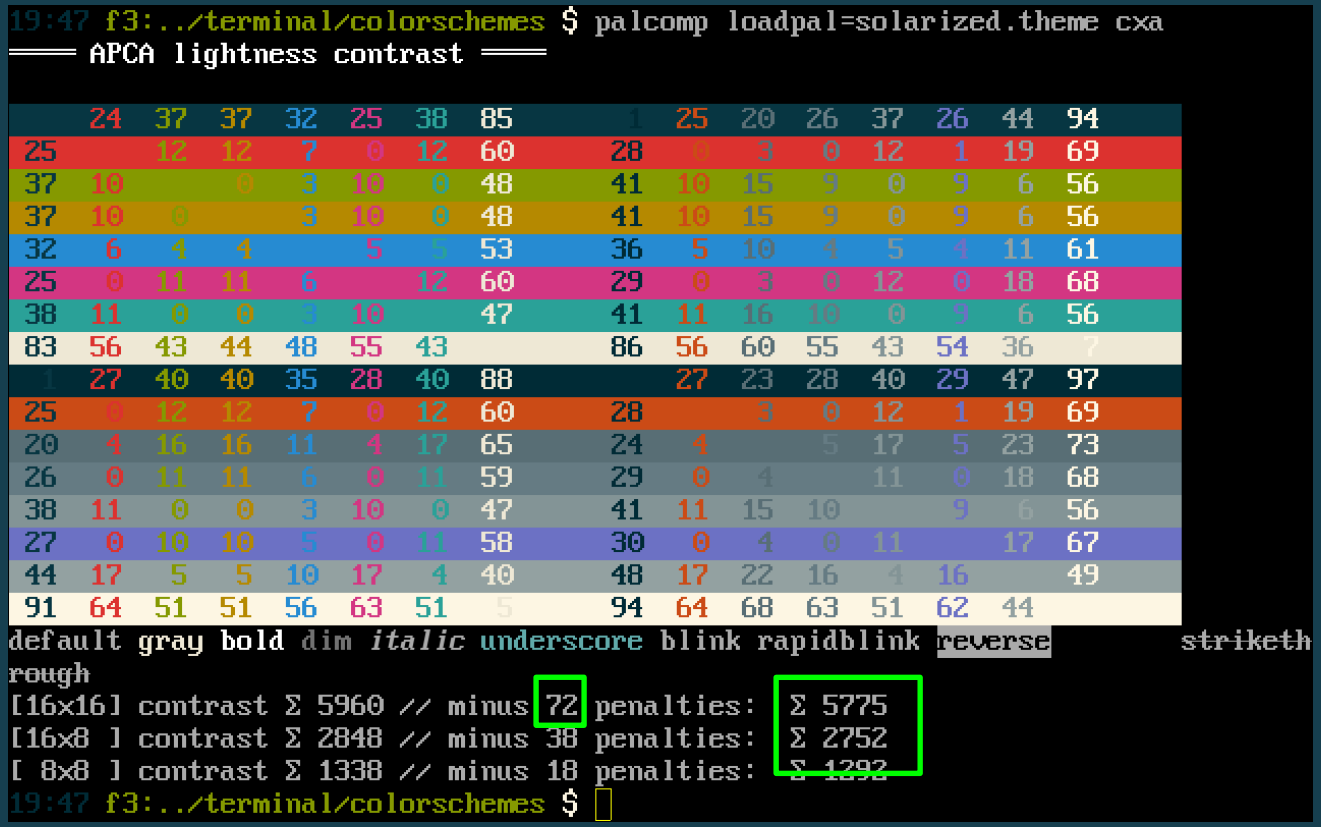
# Contrast analysis

- contrast is generally computed exclusively based on lightness (no hues)
- compute lightness: convert RGB to CIELAB (modern & perceptual), or YIQ (archaic)
- compute legibility: APCA W3 (modern & perceptual), or $abs(L_x\text{-}L_y)$ (trivial)

# Contrast analysis

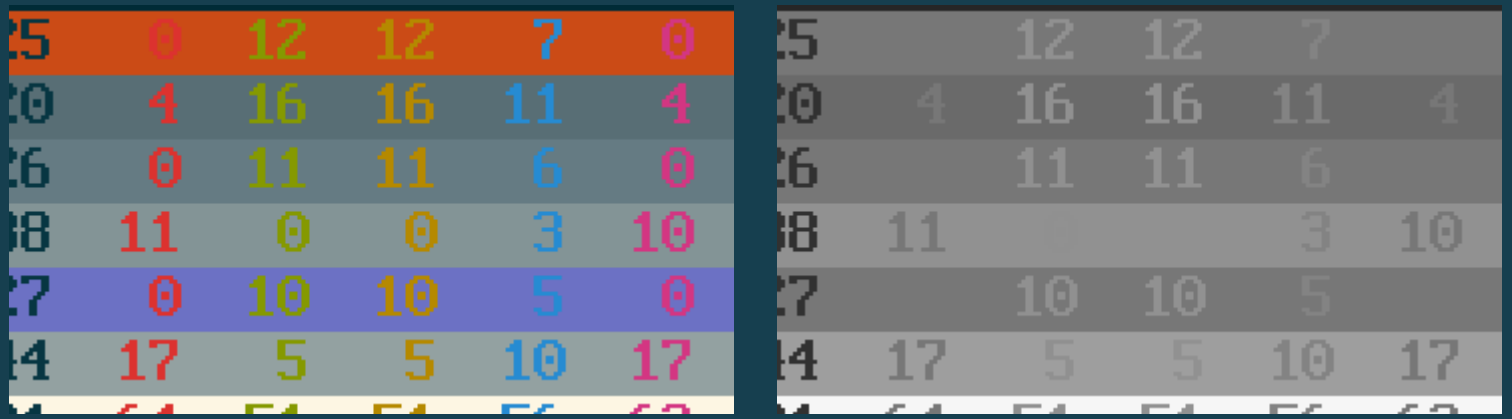
```
19:47 f3:../terminal/colorschemes $ palcomp loadpal=vga.theme cxa
═══ APCA lightness contrast ═══

    19  47  29   7  24  50  59      20  46  89 103  30  53  94 108
21      27   9  11   4  30  39       0  27  69  84  10  34  74  88
49  26      15  39  21   2  11      25   0  41  56  14   6  46  61
32   9  17      22   4  20  29       8  16  59  73   0  23  64  78
 8  11  39  21      16  42  51      12  38  81  95  22  45  86 100
26   3  22   5  16      25  34       2  22  64  79   6  29  69  83
52  29   1  18  42  24      8      28   2  38  53  18   3  43  57
60  38  10  27  51  33   7          37  10  23  44  26   3  34  48
22   0  26   9  12   3  29  38          26  68  83   9  33  73  87
48  26   0  15  39  21   3  12      25      42  56  14   6  47  61
88  66  38  55  79  61  35  26      65  38      13  54  31   3  17
102 79  51  68  92  74  48  39      78  51  11      67  44   6   2
32  10  16   0  22   4  19  28       9  15  58  72      23  63  77
55  33   5  22  45  27   1   5      32   5  34  49  21      40  54
93  70  42  60  83  65  39  31      69  43   2   7  59  36      12
106 83  55  73  96  78  52  43      82  56  15   2  72  49  10

default gray bold dim italic underscore blink rapidblink reverse          striketh
rough
[16x16] contrast Σ 9061 // minus 33 penalties:  Σ 8960
[16x8 ] contrast Σ 4082 // minus 17 penalties:  Σ 4029
[ 8x8 ] contrast Σ 1385 // minus 8 penalties:   Σ 1353
19:47 f3:../terminal/colorschemes $
```

# Contrast analysis




- 72 fg-bg pairs considered to be illegible

# Helmholtz–Kohlrausch effect




- Some fg–bg pairings reportedly have ratings of e.g. 7, 4 or even 0 despite being legible to some viewers
- If you look at it in grayscale, the ratings are justified though

# Analyzing all the styles

- A whole lot of samples: https://github.com/termux/termux-styling
- Contrast analysis based on APCA:

```
for i in *.properties; do
        palcomp loadpal=$i cxa;
done
```

# The results are in

- almost every style has less contrast than VGA

| 16×16 | 16×8 | Name |
|---|---|---|
| 5775 | 2752 | solarized |
| 6701 | 3274 | gnome |
| 8533 | 4089 | termux:zenburn |
| 8563 | 4323 | termux:argonaut |
| 8960 | 4029 | Standard EGA/VGA |
| 9035 | 3970 | Saturated VGA (e.g. #ff0000 instead of #ff5555) |
| 9349 | 4237 | Windows 3.0 (termux:e-ink-color) |

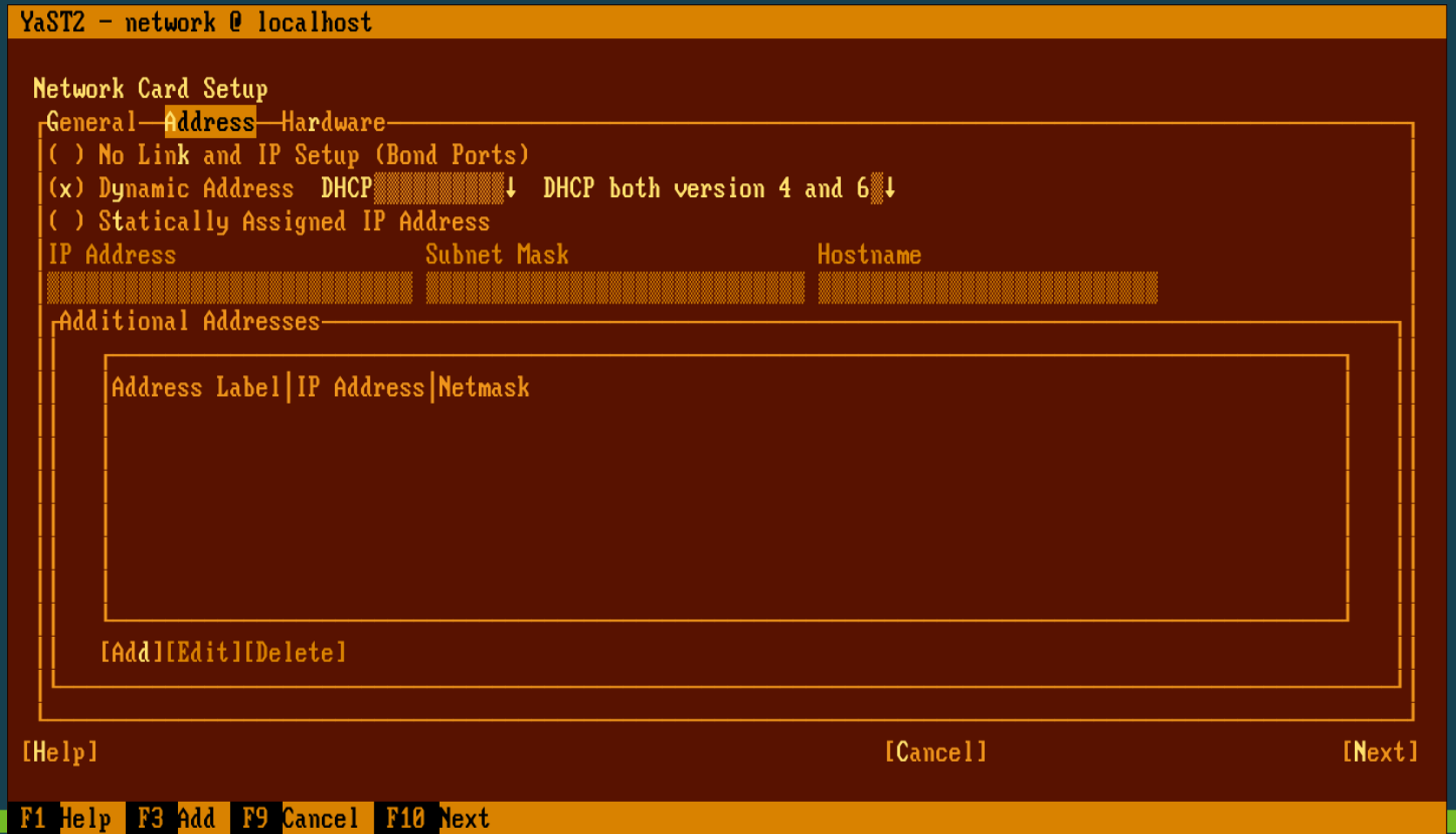
# “It's a feature not a bug”

- Low contrast in palettes sometimes advertised as a *design principle* (“for night-time coding”)
- Terminal programs provide no information about the palette's intent
- Preset files may have a description in them
- We need to distinguish between *application-specific* and *general-purpose* (GP) palettes

# GP-palette requirements

- Programs *do not* know what palette is used by the terminal
- Broad assumption is that `\e[32m` produces some green
- ECMA-48 indeed defines the general hue that escape codes ought to produce; no requirements on saturation, lightness or contrasts
- Some palettes can lead to readable text even without adhering to the hue specification (such as a hue-rotated VGA palette, or monochromatic palette)

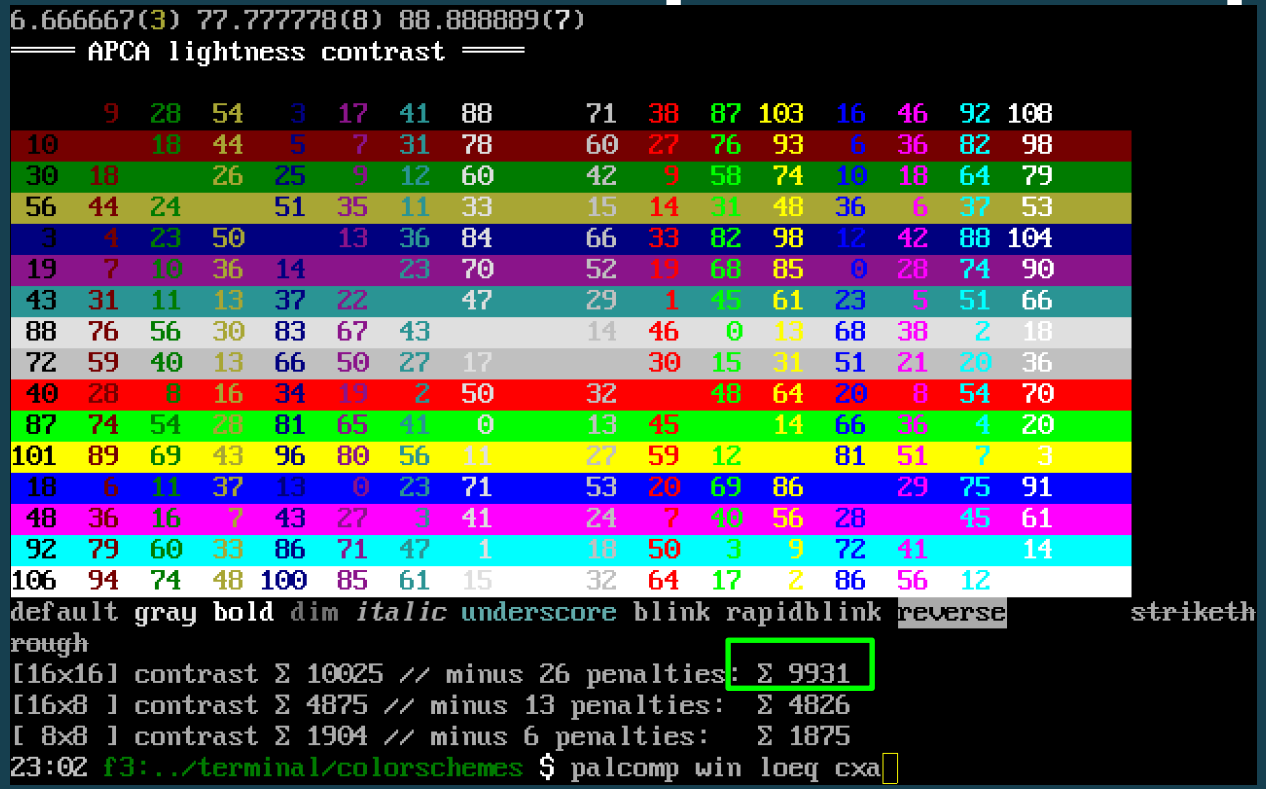
# Monochromatic/amber CRT look




# Towards an optimal palette

- If we equal-space the lightness of every color, *sum(abs($L_{fg}$-$L_{bg}$))* will be maximized in principle.
- *max sum(APCA($L_{fg}$,$L_{bg}$))* harder to compute, left for another day
- 16 spots to fill in the palette, but the modeled human can only distinguish 14–15 brightness levels

# Improved Win3 palette




- about 6% higher numerical contrast thanks to—in this case—equalization of *the lower 8 colors*

# Challenges

- Problems inherent in CIELAB/CIELch:
  - poor blue constancy
    (blue tones look quite purple)
  - LCh={0,100,301°}, despite no light, somehow equates to sRGB #00007f; stuff like that might be responsible for severe rounding errors
- wanted to use OKLAB, but OKLch is disabled in most builds of libbabl (poor distro penetration)

# Takeaways

- We need a contrast-y palette as a default, make no assumptions about the user base nor the programs that are going to run in the terminal
- The bog-standard palettes we had for the past 30 years (VGA / Windows) have reasonably good contrast, the GNOME/KDE default palettes are behind

# Thanks for your attention

https://codeberg.org/consoleet/consoleet-utils

A set of utilities for manipulating terminal fonts
and experimenting with color palettes.