



Nízkoúrovňové programování

Pokročilá práce s terminálem

Petr Krajča



Katedra informatiky
Univerzita Palackého v Olomouci



```
#include <sys/ioctl.h>
int ioctl(int fildes, int request, ... /* arg */);
```

- ovládání seriové linky (resp. znakových zařízení)
- „příkaz“ obsahující další „příkazy“
- nastavení vlastností vstupu (opakování výstupu, konverze signálů, zakončení řádku)
- zjištění stavu (např. velikost terminálu)

```
struct termios ctrl;
ioctl(STDIN_FILENO, TCGETS, &ctrl);
tcflag_t old = ctrl.c_lflag;
ctrl.c_lflag = ISIG;
ioctl(STDIN_FILENO, TCSETS, &ctrl);
```

- podrobnosti, viz man termios



- emulace hardwarových terminálů
- komplexní funkcionalita
- k dispozici přes escape sekvenci `\033[`

Příklady

- `\033[2J // vymazání terminálu`
- `\033[r;cH" // přesun na souřadnice r (row) a c (column)`
- `\033[93;44m // změna barvy písma`
- více např. https://en.wikipedia.org/wiki/ANSI_escape_code



```
#include <poll.h>
```

```
int poll(struct pollfd fds[], nfd_t nfd, int timeout);
```

- umožňuje pasivně čekat na události spojené se soubory (proudy dat)
- test, zda jsou na vstupu/výstupu data, zda došlo k zavěšení (HUP), apod.
- podrobnosti, viz `man 3 poll`



```
struct pollfd {
    int    fd;        // deskriptor souboru
    short  events;    // priznaky udalosti, na které čekáme
    short  revents;   // udalosti, které nastaly
};
```

```
struct pollfd pfd;
pfd.fd = STDIN_FILENO; // práce se std. vstupem
pfd.events = POLLIN;   // čekáme na vstup
int state = poll(&pfd, 1, timeout);
if (state < 0) return -1;
if (pfd.revents & POLLIN) { /* data jsou na vstupu */ }
```



- Nastudujte si escape sekvence a udělejte hru barevnou.
- (Volitelně) Hru vylepšete, např. počítání score, lepší grafika, NPC, pohyb v obou osách.
- (Volitelně) Přepište v příložené hře (alespoň dvě vámi zvolené) nízkoúrovňové funkce pro práci s terminálem do assembleru tak, aby používaly volání jádra OS.
- (Volitelně) Přepište celou hru do assembleru.