

# Informační a komunikační technologie

## Soubory v C

**David Weber**

Kabinet K13

[weber3@spsejecna.cz](mailto:weber3@spsejecna.cz)

# Co probereme...

- Jak soubory fungují
- Základní práce se soubory
- Funkce pro čtení a zápis do souboru

# Proč chceme soubory?

- Data zadaná do konzole se po ukončení programu ztrácí.  
⇒ Chceme umět data uchovat i po jeho vypnutí.
- Chceme umět **načítat** data ze souboru.  
⇒ Mnohdy rychlejší než je zadávat přímo do konzole.
- K tomu se hodí použít soubory.

# Základní informace

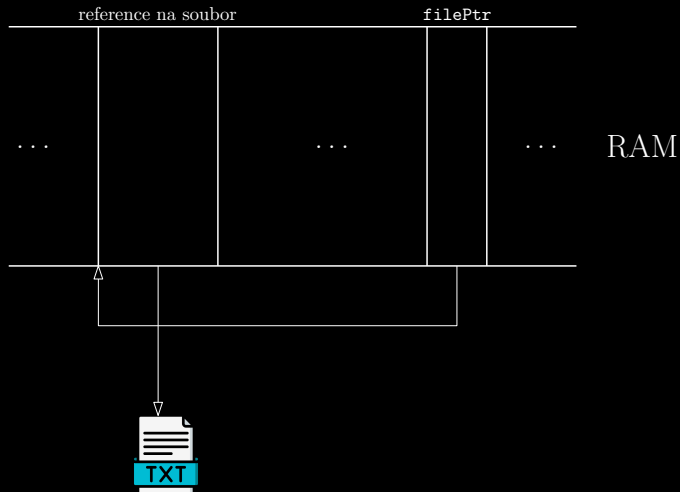
- Budeme pracovat s **textovými soubory**
  - Přípona `.txt`
- K práci se soubory používáme v jazyce C **ukazatel typu FILE**.  
`FILE* filePtr;`
- $\Rightarrow$  `filePtr` je ukazatel na referenci na konkrétní soubor.

# Otevření souboru I

- K otevření souboru používáme funkci `fopen()`.  
⇒ **Tip:** všechny funkce pro manipulaci se soubory začínají na písmeno `f` (od slova *file* = *soubor*).
- Funkce přijímá dvojici parametrů (obojí si později přiblížíme 😊):
  - **název souboru,**
  - **režim otevření souboru.**

```
FILE* filePtr = fopen("soubor.txt", "r");
```

# Otevření souboru II



# Zavření souboru I

- Každý program má v rámci systému omezené prostředky  
⇒ v jeden moment může být otevřen pouze omezený počet souborů.
- Soubor je potřeba po otevření také někdy uzavřít.
- ⇒ K tomu slouží funkce `fclose()`
- Ta přijímá jako (jediný) parametr **ukazatel na daný soubor**.

```
FILE* filePtr = fopen("soubor.txt", "r");  
...  
fclose(filePtr);
```

# Zavření souboru II

Soubory uzavíráme vždy! Je to dobrá praktika. 😊



# Nulový ukazatel

- Nulový ukazatel (angl. *NULL pointer*) je ukazatel na adresu 0x00.
- Používá se (mimo jiné) k detekci chyb v programu.
- K inicializaci nulového ukazatele lze použít makro `NULL` (nebo hodnotu 0).

```
int* ptr = NULL;
```

# Zpět k funkci fopen()

- Funkce vrací jako návratovou hodnotu adresu reference na daný soubor.
- Co když se ale z nějakého důvodu nepodaří soubor otevřít? (Např. daný soubor neexistuje.)

⇒ Funkce vrátí hodnotu NULL!

- V programu tak můžeme snadno zjistit, zda jsme úspěšně soubor otevřeli:

```
FILE* filep = fopen("text.txt", "r");  
if (filep == NULL) {  
    printf("Chyba pri otvirani souboru.");  
    return -1;  
}
```

# Režimy otevření souboru I

- Soubor lze v základu otevřít ve třech režimech:
  - pro čtení (`read`),
  - pro zápis (`write`),
  - pro připsání (`append`).

# Režimy otevření souboru II

- Pro základní práci se soubory máme celkem 6 možných režimů.
- Všechny režimy vyjma `r` a `r+` vytváří nový soubor, pokud neexistuje.

Režim	Popis	Vytvoří soubor, když neex.
"r"	Pouze čtení ze souboru	NE
"w"	Pouze zápis do souboru	ANO
"a"	Pouze přidávání do souboru	ANO
"r+"	Zápis a čtení ze souboru	NE
"w+"	Zápis a čtení ze souboru	ANO
"a+"	Přidávání a čtení ze souboru	ANO

# Funkce pro manipulaci se soubory

- Již známe základní funkce `fopen()` a `fclose()`.
- Další funkce:
  - `fscanf()`, `fgets()` – čtení ze souboru,
  - `fprintf()` – zápis do souboru,
  - `fseek()`, `rewind()` – pohyb kurzoru na určité místo v souboru.

# Funkce fscanf()

- Slouží ke čtení ze souboru do proměnné/proměnných.
- Stejně jako klasický scanf(), který již známe. 😊
- Přijímá argumenty:
  - **ukazatel na soubor,**
  - **formátovací řetězec,**
  - **ukazatele na proměnné.**
- Vrací **počet úspěšně načtených argumentů.**

```
FILE* filep = fopen("text.txt", "r");
```

```
char str[20];
```

```
int number;
```

```
int paramCount = fscanf(filep, "%s %d", str, &number);
```

# Funkce fgets() I

- Podobně jako fscanf(), i funkce fgets() čte text ze souboru.
- Načtený řetězec však nijak neformátuje, **ukládá jej jako string**.
- Oproti fscanf() čte i mezery.
- Argumenty funkce:
  - **ukazatel na string (kam načítáme),**
  - **počet znaků k přečtení,**
  - **ukazatel na soubor.**

```
FILE* filep = fopen("text.txt", "r");
```

```
char str[20];  
fgets(str, 20, filep);
```

# Funkce fgets() II

- Funkce vrací `NULL` pointer při neúspěchu, jinak vrací **ukazatel na string**.
- Návratový datový typ je tedy `char*`.
- $\Rightarrow$  To lze využít pro detekci chyby.

```
FILE* filep = fopen("text.txt", "r");
char str[20];
char* success = fgets(str, 20, filep);
if (success == NULL) {
    printf("Nepodarilo se nacíst ze souboru.");
    exit(-1);
}
```



# Funkce fprintf() I

- Formátovaný zápis do souboru.
- Funguje stejně, jako klasický printf().
- Argumenty:
  - **ukazatel na soubor,**
  - **(formátovaný) řetězec,**
  - **případně další proměnné.**

```
FILE* filep = fopen("text.txt", "w");
```

```
// Zapiše "Hello World!" do souboru  
fprintf(filep, "Hello World!");
```

# Funkce fprintf() II

```
FILE* filep = fopen("text.txt", "w");  
  
int input;  
scanf("%d", &input);  
  
// Zapise zadane cislo do souboru  
fprintf(filep, "Input number: %d", input);
```

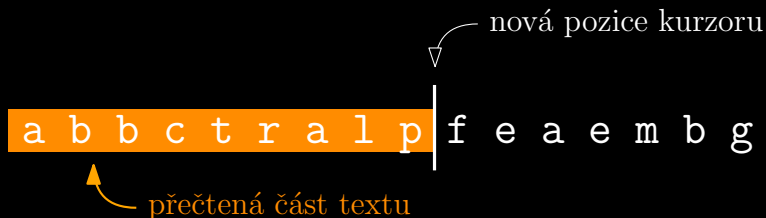
# Funkce fprintf() III

- Vrací počet zapsaných znaků do souboru při úspěchu, jinak vrací  $-1$ .
- To samé vrací i printf().

```
FILE* filep = fopen("text.txt", "w");  
int charCount = fprintf(filep, "Hello World!");  
if (charCount == -1) {  
    printf("Chyba pri zapisovani do souboru.")  
    exit(-1);  
}
```

# Pozice kurzoru v souboru

- Už jsme si vysvětlili, že FILE\* je ukazatel na místo v paměti, kde se nachází **reference na otevřený soubor**.
- Tam se (mimo jiné) ukládá také **aktuální pozice v souboru**.
- Vždy, když přečteme/zapišeme do souboru nějaký text, pozice kurzoru se posune.



# Funkce `fseek()` I

- Funkce pro posun kurzoru na určitou pozici.
- Argumenty:
  - **ukazatel na soubor**,
  - **offset (posunutí kurzoru)**,
  - **odkud offset počítat**.
- $\Rightarrow$  `fseek(<ukazatel>, <offset>, <odkud>);`
- Offset může být *kladné*, *nulové* i *záporné* číslo.

# Funkce `fseek()` II

- Offset lze počítat celkem ze tří možných míst:
  - od začátku textu → makro `SEEK_SET`,
  - od aktuální pozice kurzoru → makro `SEEK_CUR`,
  - od konce textu → makro `SEEK_END`.

| a b b c t r a l p | f e a e m b g |

`SEEK_SET` `SEEK_CUR` `SEEK_END`

# Funkce `fseek()` – `SEEK_CUR`

Počítání offsetu 2 od *aktuální pozice kurzoru*.

```
fseek(filep, 2, SEEK_CUR);
```



# Funkce `fseek()` – `SEEK_CUR`

Počítání offsetu  $-2$  od *aktuální pozice kurzoru*.

```
fseek(filep, -2, SEEK_CUR);
```

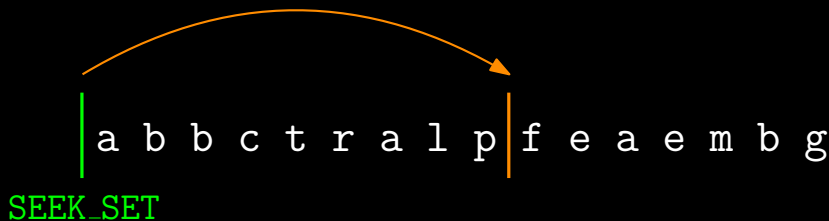




# Funkce fseek() – SEEK\_SET

Počítání offsetu 9 od *počítací pozice*.

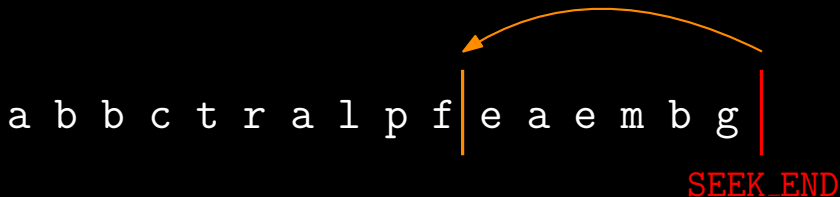
```
fseek(filep, 9, SEEK_SET);
```



# Funkce fseek() – SEEK\_END

Počítání offsetu od *koncové pozice*.

```
fseek(filep, -6, SEEK_END);
```



# Funkce rewind()

- Přesune kurzor na začátek souboru.
- Zkratka za

```
fseek(filep, 0, SEEK_SET);
```

- Jako argument přijímá pouze **ukazatel na soubor**.

```
FILE* filep = fopen("text.txt", "r");  
char buffer[20];  
fgets(buffer, 20, filep);  
  
// Presune kurzor na zacatek souboru  
rewind(filep);
```

# Otázky?

