Téma 17: Práce se soubory

Pro práci se soubory slouží třída **File**. Zároveň slouží jako manažer souborů. Na začátku programu je nutné importovat knihovnu java.io.*.

Instanci třídy File lze vytvořit třemi způsoby.

- `File soubor = new File([absolutni_cesta], [jmenoSouboru])`
- `File soubor = new File([relativni_cesta]/[adresar]/[jmen_souboru])`
- `File soubor = new File([jmeno_souboru])`

Ovšem nesmíme zapomenout, na jakém **operačním systému** pracujeme, pro Unix používáme lomítka (/) a pro Windows používáme lomítka (\).

Zde jsou vyjmenované **metody třídy File**.

Metoda	Return	Činnost
exist()	boolean	Zjistí zda soubor existuje
isFile()	boolean	Zjistí zda je soubor nebo adresář
isDirectory()	boolean	Zjistí zda je adresář nebo soubor
canRead()	boolean	Zjistí zda jde ze souboru číst
canWrite()	boolean	Zjistí zda do souboru lze zapisovat
createNewFile()	boolean	Pokud se soubor podaří vytvořit, vrátí true
mkdir()	boolean	Pokud se adresář podaří vytvořit, vrátí true
delete()	boolean	Vrátí true, pokud se podaří soubor nebo adresář zrušit
renameTo(File dest)	boolean	Vrátí true, pokud se podaří soubor nebo adresář přejmenovat
length()	long	Vrátí velikost souboru v bajtech
lastModified()	long	Vrátí systémový čas od poslední modifikace
getName()	String	Vrátí jméno souboru
getPath()	String	Celá cesta k souboru
getAbsolutePath()	String	Vrátí absolutní cestu
getParent()	String	Vrátí jméno adresáře, ve kterém je soubor obsažen
list()	String[]	Vrací pole názvů souborů a podadresářů daného adresáře

Pokud chci v Javě pracovat s nějakým souborem, ať už textovým nebo binárním, vždy musím udělat tzv. **Stream**. Stream v angličtině znamená potok nebo říčka, a vlastně to perfektně vystihuje jeho vlastnost v Javě. Stream je proud dat z programu na nějaké místo na disku, konkrétně k souboru, s nímž pracuji.

Stream může být vstupní pro čtení souboru, nebo výstupní pro zápis do souboru, existuje i varianta, že mohu číst a zapisovat do souboru pomocí jednoho proudu.

Streamy lze dál dělit dle toho, jaké data v něm tečou a to na proudy bajtové, proudy znakové, tyto dva jsou důležité, pak jsou také proudy z vyrovnávací paměti, proudy datové, standardní proudy a objektové proudy. Ale ty nás nemusí zas tak zajímat.

Pokud s proudem přestanu pracovat, je důležité ho zavřít, a to hlavně výstupní proud, jelikož se jináč zapsaná data neuloží.

Základní třídy pro práci s textovými soubory jakou jsou *.txt, *.java, *.sql a mnoho dalších jsou **java.io.FileReader** pro čtení a **java.io.FileWriter** pro zápis. Obou třídám v konstruktoru mohu předat buď objekt třídy File, nebo řetězec s cestou k souboru. Při vytváření instance třídy FileReader může nastat výjimka java.io.FileNotFoundException pokud soubor neexistuje. Pokud soubor neexistuje a vytvářím instanci třídy FileWriter, tak se soubor vytvoří, ale může nastat java.io.IOException pokud soubor nelze vytvořit.

Instance FileReader()

```
FileReader fr = new FileReader(soubor);
```

Metoda	Return	Činnost
ready()	boolean	Zjistí zda ze souboru lze ještě číst. Využití ve while cyklu
read()	int	Přečte znak a vrátí jeho hodnotu v ASCII kódu.*
close()	void	Ukončí práci a zavře proud

*Při výpisu znaku je nutno převést číslo v ASCII kódu na znak

```
int znak = ctecka.read();
System.out.format("%c", znak);
```

Instance FileWriter()

```
FileReader fw = new FileWriter(soubor);
```

Metoda	Return	Činnost
write(String)	void	Zapíše do souboru celý řetězec
write(char)	void	Zapíše do souboru znak, jehož ASCII hodnotu mu předáme
close()	void	Ukončí práci a uloží soubor. Tato metoda je hodně důležitá

Java nabízí dva základní a velice sympatické **proudy s vyrovnávací pamětí**. Třída java.io.BufferedReader pro čtení a třídu java.io.BufferedWriter pro zápis. Práce je s nimi obdobná jako s FileReader a FileWriter. S tím že do konstruktoru se nepředává řetězec s cestou, ale instance třídy FileReader nebo FileWriter. Obdobně je to s BufferedWriterem.

Hlavní rozdíl mezi normálním proudem a proudem s vyrovnávací pamětí je ten že proudy bez vyrovnávací paměti pokud potřebují přistoupit k souboru, musí se spojit přímo s diskem, síťovou jednotkou a tak, kdežto proudy s vyrovnávací pamětí si načtou data do vyrovnávací paměti a k zařízení přistupují jen tehdy, když je vyrovnávací paměť v případě readru prázdná, nebo v případě writeru plná.

Metody BufferedReader()

Metoda	Return	Činnost
read()	int	Přečte znak, jenž je na řadě a vrátí ASCII hodnotu.
readLine()	String	Přečte celý řádek, a vrátí v podobě řetězce.

skip(long)	void	Poskočí dopředu o požadovaný počet bajtů
close()	void	Zavře stream

Metody BufferedWriter()

Metoda	**Return**	**Činnost**
write(String)	void	Zapíše do souboru řetězec
write(int)	void	Zapíše do souboru znak odpovídající hodnotě v ASCII tabulce
newLine()	void	Zapíše do souboru odřádkování
flush()	void	Vyprázdní vyrovnávací paměť
close()	void	Zavře stream a uloží soubor