

# Soubory

Tomáš Pitner, Radek Ošlejšek, Marek Šabo

# Hlavní balíky V/V operací

- Základní věci jsou v balících `java.io`, `java.nio.file`.
- Základem je třída `java.io.File`

## Práce se soubory přes objekty File

- Objekt třídy `File` je de facto nositelem jména souboru, jakási "brána" k fyzickým souborům na disku.
- Nejde tedy o datovou strukturu nesoucí např. obsah souboru.
- Používá se jak pro soubory, tak *adresáře*, *linky* i soubory identifikované *UNC jmény*
- Je plně platformově nezávislé.

## Odlišnosti systémů souborů

Na odstínění odlišností jednotlivých systémů souborů lze použít vlastností (uvádíme jejich hodnoty pro JVM pod systémem MS Windows):

*File.separatorChar*

`\\` jako char

*File.separator*

totéž jako String

*File.pathSeparatorChar*

`:` jako char

*File.pathSeparator*

totéž jako String

*System.getProperty("user.dir")*

adresář uživatele, pod jehož UID je proces JVM spuštěn

## Vytvoření objektu File

Pro vytvoření objektu třídy `File` konstruktorem (NEJEDNÁ SE PŘÍMÉ VYTVOŘENÍ SOUBORU NA DISKU!) máme několik možností:

*new File(String \_filename\_)*

zpřístupní v aktuálním adresáři soubor s názvem *filename*

*new File(File \_baseDir\_, String \_filename\_)*

zpřístupní v adresáři *baseDir* soubor s názvem *filename*

*new File(String \_baseDirName\_, String \_filename\_)*

zpřístupní v adresáři se jménem *baseDirName* soubor s názvem *filename*

*new File(URL \_url\_)*

zpřístupní soubor se souborovým (file:) URL *url*

## Existence a povaha souboru

*boolean exists()*

vrátí true, právě když zpřístupněný soubor (nebo adresář) existuje

*boolean isFile()*

test, zda jde o soubor a nikoli adresář

*boolean isDirectory()*

test, zda jde o adresář

## Přístupová práva k souboru

*boolean canRead()*

mám právo čtení souboru?

*boolean canWrite()*

mám právo zápisu souboru?

## Vytvoření souboru/adresáře

*boolean createNewFile()*

zkusí vytvořit soubor *soubor* a vrací true, právě když se podaří vytvořit.

*boolean mkdir()*

obdobně pro adresář

*boolean mkdirs()*

navíc si umí dotvořit i příp. neexistující adresáře na cestě

## Vytvoření dočasného souboru

*static File createTempFile(String \_prefix\_, String \_suffix\_)*

Vytvoření dočasného (temporary) souboru — skutečně fyzicky vytvoří dočasný soubor ve standardním, pro to určeném, adresáři (např. c:/temp) s uvedeným prefixem a sufixem názvu

*static File createTempFile(String \_prefix\_, String \_suffix\_, File \_directory\_)*

dtto, ale vytvoří dočasný soubor v zadaném adr. *directory*

# Smazání, přejmenování

*boolean delete()*

zrušení souboru nebo adresáře

*boolean renameTo(File \_dest\_)*

prejmenuje soubor nebo adresář (neumí přesun souboru/adresáře)

## Další vlastnosti

*long length()*

délka (velikost) souboru v bajtech

*long lastModified()*

čas poslední modifikace v ms od začátku éry — tj. ve stejných jednotkách a škále jako systémový čas vrácený `System.currentTimeMillis()`.

*String getName()*

jen jméno souboru (tj. poslední část cesty)

*String getPath()*

celá cesta k souboru i se jménem

*String getAbsolutePath()*

absolutní cesta k souboru i se jménem

*String getParent()*

adresář, v němž je soubor nebo adresář obsažen

- Blíže viz [dokumentace API třídy File](#).

## Práce s adresáři

- Klíčem je opět třída `File`, použitelná i pro adresáře
- Jak např. získat (filtrovaný) seznam souborů v adresáři?
- Pomocí metody `File[] listFiles(FileFilter ff)` nebo podobné `File[] listFiles(FilenameFilter fnf)`.
- `FileFilter` je rozhraní s jedinou metodou `boolean accept(File pathname)`
- obdobně `FilenameFilter`
- Viz [Popis API java.io.FilenameFilter](#).