Jar – vytváření a používání archivů

Formát souborů JAR (Java Archive), používaný pro seskupování více souborů (většinou souborou s příponou .class, ale i dalších) do jednoho zkomprimovaného, je odvozen z klasického formátu archivů ZIP. Soubory JAR jsou přenositelné mezi jednotlivými platformami a mohou obsahovat další doplňující informace o vložených souborech – používá k tomu tzv. „manifest“ soubor (v archivu je uložen pod jménem META-INF\MANIFEST.MF) a případně soubory s digitálními podpisy (META-INF*.SF). Soubory JAR se vytvářejí pomocí programu jar, který je nedílnou součástí distribuce Java SDK.

Vytváření archivu

Program jar má parametry příkazové řádky odvozené od archivačního program tar, který se používá v unixových operačních systémech. Syntaxe volání nástroje jar vypadá následovně:

jar [volby] archiv vstupní_soubor[y]Přípustné volby jsou následující:

parametr	**význam**
c	bude se vytvářet nový archiv (create)
u	existující archiv se bude aktualizovat (přidávat či měnit soubory)
t	vypíše se obsah existujícího archivu (table of contents)
x	soubory z archivu se zapíší do aktuálního adresáře a případně do podadresářů (extract)
x soubor	z archivu se vypíše pouze zadaný soubor
f	je zadán vstupní/výstupní archiv (jeho jméno je nutné uvádět i s koncovkou .jar), pokud není uveden tento parametr, použije se standardní vstup (s parametrem x či t) nebo standardní výstup (při vytváření)
v	na obrazovce zobrazuje podrobněji svoji činnost (verbose)
0	soubory se pouze vkládají, nekomprimují se
m manifest	do archivu vloží uživatelem vytvořený a v příkazové řádce určený soubor manifest
M	nebude se vytvářet soubor MANIFEST.MF

tabulka 3 Parametry programu jar

Příklady použití:

Vytvoření archivu:

```
jar cvf mujArchiv.jar *.class
jar cvf it_112.jar cz\vse\it_112\*.class
```

Aktualizace archivu

```
jar uvf it_112.jar cz\vse\it_112\*.class
```

Výpis obsahu archivu

```
jar tvf it_112.jar
```

Extrahování souboru z archivu do aktuálního adresáře a podadresářů

```
jar xvf it_112.jar
```

Vytvoření archivu včetně vlastního manifest souboru

```
jar cvmf mujManifest.mf mujArchiv.jar *.class
```

Do jar archivu lze vkládat (a z něho používat) i obrázky a zvukové soubory.

Zpřístupnění jar archivu pro JVM

Pro používání jar archivu v aplikaci je nutno soubor jar specifikovat v proměnné CLASSPATH (nebo v parametru classpath příkazů javac a java) pro vyhledávání tříd (souborů .class). Nestačí aby byl archiv v aktuálním adresáři.

Příklad nastavení CLASSPATH (ve Windows):

```
set CLASSPATH= C:\archivy\mujArchiv.jar;C:\archivy\it_112.jar;.
```

- Java bude vyhledávat soubory .class i v archivech mujArchiv.jar, it_112.jar, uložených v adresáři archivy na disku C, a v aktuálním adresáři (a samozřejmě ve standardním archivu jre\lib\rt.jar).

Kód appletu může být také představován více třídami a je možné (a vhodné) ho zabalit do archivu. Používání archivu musí být zapsáno v tagu appletu (parametr archive).

Následují dva příklady použití souborů jar v appletu:

```
<applet code=Animator.class
archive="jars/animator.jar"
width=460 height=160>
<param name=foo value="bar">
</applet>

<applet code=Animator.class
archive="classes.jar, images.jar, sounds.jar"
width=460 height=160>
<param name=foo value="bar">
</applet>
```

Úpravy souboru manifest.mf

Archiv jar obsahuje soubor manifest.mf v adresáři META-INF. Tento soubor obsahuje základní informace o archivu pro JVM.

Zde si popíšeme, jak vytvořit spustitelný archiv.

Pokud máte aplikaci zabalenou v souboru JAR, potřebujete v souboru Manifest.mf označit třídu, která se má spouštět (tj. třídu, ze které se má spustit metoda *public static void main (String[] args)*). Toho se docílí řádkem:

```
Main-Class: třída
```

Tento řádek doplníme na konec manifest souboru a znovu ho zabalíme do archivu. Po dopsání této informace je třeba odřádkovat, jinak není manifest soubor v JVM správně interpretován.

Vlastní spuštění aplikace poté proběhne následujícím příkazem

```
java –jar archiv.jar
```

Pokud balíte do jar archivu aplikaci používající více archivů (např. archiv s ovladačem databáze, archiv pro připojení helpu nebo archiv s obsahem helpu), měl by být postup následující :

- Zabalíme soubory class aplikace do samostatného archivu např. aplikace.jar, a to včetně adresářové struktury naších balíčků a případně i souborů s obrázky, parametry atd. Tento archiv neobsahuje ostatní používané jar archivy. Pozor, jestliže v aplikaci používáte

parametrické soubory nebo obrázky, po zabalení do archivu je aplikace nenajde. Je třeba s tím počítat předem a kód upravit viz podkapitolka Čtení souboru uvnitř jar archivu.

- Připravíme si adresářovou strukturu celé distribuce např. takto:
  docs – adresář, který bude obsahovat dokumentaci
  sources.jar – pokud poskytujeme i zdrojové kódy, měli by být zabaleny v samostatném jar archivu
  adresarAplikace – adresář obsahující :
    aplikace.jar
    ovladac.jar
    jh.jar
    help.jar
  README.TXT - soubor se základním popisem aplikace
- Upravíme manifest soubor tohoto archivu, např. takto:

```
Manifest-Version: 1.0
Created-By: 1.4.1_03 (Sun Microsystems Inc.)
Class-Path: jh.jar mysql.jar
Main-Class: HlavniTridaAplikace
```

  Všimněte si, že jména jednotlivých archivů jsou od sebe oddělena pouze mezerami, při použití středníku jako oddělovače nefunguje nastavení CLASSPATH z archivu správně. Archiv s obsahem nápovědy není nutno uvádět (není v něm spustitelný kód), odkaz na něj musí být správně uveden v kódu aplikace.
- Celou distribuci potom zabalíme do souboru s koncovkou zip.

Čtení souboru uvnitř archivu

Pokud například používáte obrázky na tlačítka, bez jar archivu můžete mít napsané načtení obrázku takto:

```
ImageIcon obrazek = new ImageIcon("./obrazky/otaznik.gif");
```

Jestliže počítáte se zabalením aplikace do jar archivu použijte metodu getResource(String soubor) ze třídy Class. Tato metoda vezme cestu, ze které načetla kód třídy, upraví ji podle parametru a vrátí URL tohoto souboru nebo null, pokud ho nenajde. Pro vytvoření obrázku pak použijte konstruktor, který jako parametr bere URL souboru. Kód pak bude vypadat následovně:

```
java.net.URL  urlObrazku = this.getClass().getResource("./obrazky/
otaznik.gif ");
ImageIcon obrazek = new ImageIcon(urlObrazku);
```

Pokud se jedná o soubor např. s parametry, který potřebujete otevřít pro čtení, je možné použít metodu getResourceAsStream(String soubor), která vrací instanci InputStreamu (tj. soubor přímo otevře pro čtení po bytech). Čtení souboru s parametry by pak mohlo vypadat takto:

```
BufferedReader soubor = new BufferedReader (new InputStreamReader
(this.getClass().getResourceAsStream("./parametry.txt")));
```