Objektovo orientované programovanie

(balíky)

4. prednáška (2. časť)

Vladislav Novák
FEI STU v Bratislave
7.10.2014

Obsah

Balík (package)

Balík (*package*) je zoskupenie súvisiacich referenčných typov (triedy, rozhrania, enum), ktoré poskytuje ochranu prístupu a zabraňuje konfliktu názvov.

Príklady balíkov:
Základné triedy sa nachádzajú v balíku `java.lang`
Triedy pre čítanie a zápis (pre vstup a výstup) sú v balíku `java.io`.

Výhody zoskupenia typov do balíka
- typy v jednom balíku spolu súvisia
- ak napr. majú dve triedy rovnaký názov, dajú sa rozlíšiť podľa balíka, ktorého sú súčasťou
- prístupové práva k typom vo vnútri vlastného balíka môžu byť väčšie ako prístupové práva k typom iného balíka

Vytvorenie balíka

Najprv treba určiť názov balíka.
Na začiatok každého zdrojového súboru patriaceho do určitého balíka treba napísať príkaz

```
package nazovbalika;
```

Príkaz `package` musí byť prvým príkazom v zdrojovom súbore. V každom súbore môže byť len jeden príkaz `package`.

Prístupové práva

Ak je v jednom zdrojovom súbore umiestnených viacero typov (nie vnorených), potom iba jeden z nich môže byť označený ako `public` a musí mať rovnaký názov ako je názov súboru.

Typy označené ako `public` sú prístupné z ľubovoľného balíka, typy bez označenia `public` t.j. *package-private* sú prístupné iba z vnútra balíku (iné označenie nevnorených typov nie je možné).

Balík bez názvu

Ak nie je použité kľúčové slovo `package`, potom je typ umiestnený do balíka bez názvu (*default package*). Balík bez názvu je vhodný iba pre malé, alebo dočasné aplikácie.

Príklad definovania tried v balíku

Vytvorenie balíka s názvom `grafika`, ktorý obsahuje triedy `Grafika`, `Obdlznik`, `Kruh` a rozhranie `Presuvatelny`:

súbor Grafika.java:
```
package grafika;
public class Grafika{
    //....
}
```

súbor Presuvatelny.java:
```
package grafika;
public interface Presuvatelny{
    //....
}
```

súbor Obdlznik.java:
```
package grafika;
public class Obdlznik {
    //....
}
```

súbor Kruh.java:
```
package grafika;
public class Kruh {
    //....
}
```

Pomenovanie typov

Jednoduchý názov typu je názov typu ktorý je umiestnený za `class`, `interface`, `enum`. Napríklad:
`Object`
`String`
`Math`
`BufferedReader`
`FileInputStream`

Plný názov (*úplný názov*) *typu* sa skladá z názvu balíka, za ktorým nasleduje bodka a jednoduchý názov typu v balíku. Napríklad:
`java.lang.Object`
`java.lang.String` (`java.lang` – názov balíka, `String` – názov triedy)
`java.lang.Math`
`java.io.BufferedReader`
`java.io.FileInputStream`

Konvencia pomenovania balíka

- názvy balíkov sa píšu iba malými písmenami, aby sa predišlo konfliktom s názvami tried, alebo rozhraní
- spoločnosti uvádzajú na začiatku názvov svojich balíkov, svoj obrátený doménový názov (napr. sk.stuba.fei.uim)
- balíky v samotnom jazyku java začínajú reťazcami `java.` alebo `javax.`
- doménový názov z internetu nemusí byť platný, ak obsahuje špeciálne znaky (napr. pomlčku), alebo sa začína číslicou. Vtedy sa pomlčka odporúča nahradiť podčiarkovníkom, pred začiatočnú číslicu doplniť podčiarkovník a pod.

Príklad názvu balíka:
`sk.stuba.fei.uim.oop.prednaska4.grafika`

Člen balíka

Typy, ktoré tvoria balík sa označujú ako *členy balíka*.

Použitie členov balíka vo vnútri balíka

Stačí používať jednoduché meno ako doteraz. Napríklad:
`Grafika`, `Obdlznik`, `Kruznica`.

Použitie členov balíka v inom balíku

Pri používaní členov balíka mimo balíka v ktorom sú definované je potrebné:
- odkazovať sa na člen balíka pomocou úplného názvu,
- alebo importovať člen balíka,
- alebo importovať celý balík

Na importovanie balíkov, alebo členov balíka sa používa kľúčové slovo `import`.
Kľúčové slovo `import` treba v súbore umiestniť za `package` (ak existuje) a pred definíciu akéhokoľvek typu.

príklad – odkazovanie pomocou úplného názvu

```
package inybalik;

public class TestovaciaTrieda{
    public static void main(String[] args) {
        grafika.Obdlznik o = new grafika.Obdlznik();
        //.....
    }
}
```

príklad – importovanie člena balíka

```
package inybalik;

import grafika.Obdlznik;
import grafika.Kruh;

public class TestovaciaTrieda{
    public static void main(String[] args) {
        Obdlznik o = new Obdlznik();
        Kruh k = new Kruh();
        //.....
    }
}
```

príklad – import celého balíka

```
package inybalik;

import grafika.*;

public class TestovaciaTrieda{
    public static void main(String[] args) {
        Obdlznik o = new Obdlznik();
        Kruh k = new Kruh();
        //.....
    }
}
```

Importovanie verejných vnorených typov

Predpokladajme že trieda `Obdlznik` ma verejné vnorené typy `Zväčšovatel` a `Vyfarbovatel`. Tieto verejné typy môžeme importovať napr. takto:

```
import grafika.Obdlznik.*;
```

Automatické importovanie

Prekladač do každého súboru automaticky importuje

- balík `java.lang`
- aktuálny balík

Zdanlivá hierarchia balíkov

V jave existujú napr. takéto balíky:
`java.awt`,
`java.awt.color`,
`java.awt.font`
Názvy týchto balíkov začínajú rovnakým reťazcom `java.awt`, aby bolo vidno že spolu súvisia. Avšak balíky `java.awt.color` a `java.awt.font` nie sú súčasťou balíka `java.awt`. Príkaz

```
import java.awt.*;
```

importuje všetky členy balíka `java.awt`, ale neimportuje členy balíkov `java.awt.color` a `java.awt.font`. Pre importovanie členov všetkých troch balíkov treba zadať:

```
import java.awt.*;
import java.awt.color.*;
import java.awt.font.*;
```

Nejednoznačné názvy

Ak sú importované dva balíky, ktoré obsahujú rovnako pomenované typy, potom sa na tieto typy treba odvolávať plným menom.

Príkaz statického importu

Slúži na importovanie statických finálnych atribútov (konštánt) a statických metód pridaním slova `static`.

Napr. kód:

```
double r = Math.cos(Math.PI*theta);
```

možno upraviť nasledovne:

```
import static java.lang.Math.PI;
import static java.lang.Math.cos;

//neskor v kode
double r = cos(PI*theta);
```

alebo

```
import static java.lang.Math.*;

//neskor v kode
double r = cos(PI*theta);
```

Príkaz statického importu treba používať opatrne. Niekedy môže zvyšovať čitateľnosť pretože netreba stále uvádzať názov triedy (dlhý matematický vzorec). Ak je ale tento príkaz použitý príliš často, potom znižuje čitateľnosť, pretože sťažuje určovanie, ku ktorej triede statický člen patrí.

Správa zdrojových a výstupných súborov

Mnohé implementácie platformy Java, ukladajú súbory v systéme súborov podľa názvu balíka a názvu typu nasledovne:
Zdrojový súbor triedy `Obdlznik` v balíku
`sk.stuba.fei.uim.oop.prednask4.grafika` bude uložený v súbore:
<cesta k **zdrojovým** suborom>/sk/stuba/fei/uim/oop/prednask4/grafika/Obdlznik.**java**
Skompilovaný súbor bude uložený ako
<cesta k **výstupným** suborom>/sk/stuba/fei/uim/oop/prednask4/grafika/Obdlznik.**class**

Oddelenie zdrojových a skompilovaných súborov umožňuje ľahké zverejnenie skompilovaných súborov bez zdrojových súborov.

Cesta k skompilovaným súborom sa nachádza v systémovej premennej `CLASSPATH` (windows aj unix). Táto premenná môže obsahovať aj viacero ciest oddelených bodkočiarkou (windows), alebo dvojbodkou (unix).

Pri spustení programu sa hľadá skompilovaná trieda v aktuálnom adresári, v súbore JAR, ktorý obsahuje triedy platformy Java a v adresároch uvedených v premennej `CLASSPATH`.