ПАКЕТИ

ЛЕКЦИОНЕН КУРС "ПРОГРАМИРАНЕ НА JAVA"

Distributed
@learning center

СТРУКТУРА НА ЛЕКЦИЯТА

- Изграждане и използване на библиотеки
- Мотивация
- Пакети

ПРОБЛЕМ

- Основно съображение в обектно-ориентирания развой
  - Разделяне на нещата, които се променят, от тези, които остават постоянни
- Това е особено важно за библиотеките
  - От една страна, потребителите на библиотеките трябва да могат да разчитат на частта, която се използва, и да знае, че няма да има нужда от преписване на кода, ако излезе нова версия
  - От друга страна, създаделят на библиотеката трябва да има свободата да прави модификации и подобрения с увереността, че кодът на клиент-програмиста няма да бъде повлиян от тези промени
- Java предоставя спецификатори за достъп, за да позволи на създаделя на библиотеката да укаже какво е достъпно за клиент-програмиста
- Все още остава въпросът, по какъв начин компонентите се свързват в един библиотечен модул
  - Това се управлява от ключовата дума **package**

JAVA-СИНТАКСИС

```
Source_text_file ::=
        [Package_declaration]
        {Import}
        {Type_declaration}

Type_declaration ::=
        Class_declaration |
        Interface_declaration .
```

КОМПИЛАЦИОННИ ЕДИНИЦИ

- Компилационна единица
  - Файл с изходен код на Java
  - Притежава име (еднакво като това на класа), завършващо с .java
  - Само един public клас
- При компилиране на .java файл
  - Получаваме изходен файл с точно същото име, но с разширение .class
  - Всеки .java файл кореспондира със съответен .class файл

РАБОТЕЩИ ПРОГРАМИ

- Една работеща програма са няколко .class файла, които могат да бъдат пакетирани и компресирани в JAR файл
  - С помощта на jar архиватора на Java
- Java интерпретаторът отговаря за откриването, зареждането и интерпретирането на всички тези файлове

ПРИМЕР: ПЪЛНА ФОРМА НА ФАЙЛА С ПЪРВИЧЕН КОД

```
package demo;

import java.util.*

class C1   {
    ...
}
```

Дефиницията на един компонент C1 съдържа:
- **package**: множество от класове с подобни задачи
- **import**: задава пакет, от който ще се използват класове

МОТИВАЦИЯ

- API е името на всички класове, доставени заедно с компилатора на Java
  - С всяка нова версия тези класове се увеличават
- Тези предварително дефинирани класове създават един организационен проблем
  - Програмистите трябва да имат възможност да индикират, че се нуждаят само от едно тяхно подмножество във всеки първичен файл

ПАКЕТИ

- Тъй като един пакет в действителност никога не се “пакетира” в един единствен файл, той може да бъде съставен от много файлове
  - За да се предотврати това, всички файлове от определен пакет се поставят в една директория
  - Т.е. използва се йерархичната файлова структура на операционната система
  - Класовете в една директория формират един пакет
- Конкатенацията на имената на директориите формират името на пакета
  - Пример: java.applet
  - Локацията на /java зависи от използваната платформа

ИМПОРТ

- Ако искаме да използваме един клас от даден пакет можем да използваме пълно квалифицирано име
  - Пример: x = java.lang.Math.sqrt(5);
- В много случаи такова използване е неудобно
- Java представя възможност за специфициране на достъп до определени класове от даден пакет посредством **import** маханизъм
  - Импортирането предоставя механизъм за управление на "пространства от имена"

ПРИМЕР ЗА ИМПОРТ

```
java.util.Date d = new java.util.Date();
java.awt.Point p = new java.awt.Point(1,2);
java.awt.Button b = new java.awt.Button();
```

```
import java.util.Date;
import java.awt.*;

...
Date d = new Date();
Point p = new Point(1,2);
Button b = new Button();
```

JAVA.LANG ПАКЕТ

- Java предполага, че класовете java.lang са винаги налични
  - Т.е. Все едно, че import java.lang.* наличен в началото на всяка програма

ВКЛЮЧВАНЕ НАШИ КЛАСОВЕ В ЕДИН ПАКЕТ

- Всички .class файлове, които съхраняваме в някаква директория, принадлежат към един и същ безименен пакет
- За да предизвикаме въвеждане на клас в един пакет е необходимо:
  - Съхранява се в съответната директория и
  - Добавяне на дефиниция на пакет в началото на файла:
    - package *package_name*
  - Трябва да бъде първият (некоментиран) ред в първичната програма

ИМА ЛИ ЗНАЧЕНИЕ КЪДЕ ЩЕ БЪДЕ ПОМЕСТЕН ЕДИН КЛАС?

- Името, което се използва за реферирane на класа

ВИДИМОСТ НА ПРОМЕНЛИВИТЕ НА ИНСТАНЦИИ

- Какво става с видимостта на променливите на инстанции
  - public
  - private
- Какво става ако не е декларирана нито като **public** нито като **private**?

По подразбиране е видима в пакета – т.е. в методите на класовете, които принадлежат към същия пакет

ВИДИМОСТ НА КЛАСОВЕ

- Един клас, деклариран като **public** е видим за всички класове
- Един клас, не деклариран като **public** е видим за всички класове в същият пакет
- Един клас, деклариран като **private**
  - Вътрешни класове

БЛАГОДАРЯ ЗА ВНИМАНИЕТО!

КРАЙ “ПАКЕТИ”

Distributed
@learning center