9

Пакеты

ОБСУЖДАЕМЫЕ ВОПРОСЫ

1. Пакеты
2. Компиляция и выполнение приложения с классами, размещенными в разных пакетах
3. Переменная **CLASSPATH**

ПАКЕТЫ

Пакеты Java обеспечивают пространства имен для классов.
Примеры названий пакетов:

```
java.lang;
java.io;
org.springframework.beans;
```

Пакеты, по-сути, являются иерархической структурой (деревом).

Обычно корневой пакет, название которого расположено слева в полном имени пакета, должен иметь глобальную уникальность. Он может совпадать с названием предприятия, фамилией известного автора книг, уникальным (по мнению разработчика) названием приложения и т.п.

ПАКЕТЫ

Для обеспечения глобальной уникальности в качестве корневого пакета Sun рекомендовал использовать доменное имя, например `sun.com`. Таким образом, глобальным пакетом для Sun будет - `com.sun`.

Пакеты нижних уровней обычно связаны с функциональным назначением классов, находящихся в данном пакете. То есть пакет, который содержит группу классов, объединяет их по некоторому смыслу.

ПАКЕТЫ (ПРОДОЛЖЕНИЕ)

То есть пакет группирует вложенные пакеты и классы по какому-либо критерию – по разработчику, по функциям и пр. В полное название класса входит названия всей цепочки пакетов в иерархии.

```
Project_DIR
   |
   |-- classes
   |       `-- zoostore
   |             |-- model
   |             |     `-- Cat.class
   |             |
   |             `-- test
   |                   `-- TestCats.class
   |
   `-- src
        `-- zoostore
              |-- model
              |     `-- Cat.java
              |
              `-- test
                    `-- TestCats.java
```

Структура директорий, в которых хранятся файлы классов, должна соответствовать структуре пакетов.

При этом желательно файлы исходных кодов и файлы классов размещать отдельно, что повышает управляемость:

ПАКЕТЫ (ПРОДОЛЖЕНИЕ)

Пакет описывается только внутри классов оператором `package`. Этот оператор должен находиться в самом начале файла исходного кода. Выше него могут быть только пустые строки и комментарии.

Теперь класс `Cat`, с помощью оператора `package`, мы можем поместить в отдельный пакет, например, `zoostore.model`.

```
package zoostore.model;
```

Класс **TestCats** будет помещен в пакет, но отдельно от класса `Cat`:

```
package zoostore.test;
```

А как же обстоит с теми классами, которые мы писали раньше и не задумывались о помещении их в какие-либо пакеты? Они находятся вне пакетов вообще? Нет, они по умолчанию помещаются в пакет с названием `default`.

ЗАМЕЧАНИЕ: Для названий пакетов принято использовать буквы нижнего регистра.

ИЗМЕНЕННЫЙ ТЕКСТ КЛАССА TESTCATS

```
package zoostore.test;

import zoostore.model.Cat;

public class TestCats {
 public static void main(String[] args) {
     Cat cat_1 = new Cat();
     cat_1.name = "Барсик";
     cat_1.weight = 5;
     cat_1.age = 3;
     Cat cat_2;
     cat_2 = new Cat();
     cat_2.name = "Мурка";
     cat_2.weight = 4;
     cat_2.age = 2;
           System.out.println("------------");
     cat_1.printDescription();
           System.out.println("------------");
     cat_2.printDescription();
           System.out.println("------------");
     cat_2 = cat_1;
     cat_2.printDescription();
  }
}
```

Оператор `import` требуется для ссылки на классы, находящиеся в других пакетах. Если вы ссылаетесь на множество классов из одного пакета, можно указать на это с помощью символа *

```
import java.io.*;
```

Помимо оператора `package` мы добавили оператор `import`, который ссылается на класс `zoostore.model.Cat`. Конечно, вместо этого мы могли бы всюду ссылаться на полностью квалифицированное имя класса `Cat`, например, так:

```
zoostore.model.Cat cat_1 = new zoostore.model.Cat();
```

Однако это снижает ясность текста, и использовать такой стиль написания следует в редких случаях.

КОМПИЛЯЦИЯ И ВЫПОЛНЕНИЕ ПРИЛОЖЕНИЯ TESTCATS

Теперь компиляция и выполнение программы потребуют некоторых дополнительных усилий. Набор выполняемых для этого команд требует дополнительного внимания:

```
C:
cd \
cd  sources\demo\d_07
SET CLASSPATH=./classes

javac -d ./classes ./src/zoostore/model/Cat.java

javac -d ./classes ./src/zoostore/test/TestCats.java
java zoostore.test.TestCats
```

С помощью команды `SET CLASSPATH=./classes` мы устанавливаем значение переменной окружающей среды операционной системы, `CLASSPATH`, которая указывает, где находятся классы приложения.

ПЕРЕМЕННАЯ CLASSPATH

Если требуется добавить ссылки на дополнительные библиотеки (JAR файлы) это делается с использованием разделителя (; для Windows), например, так:

```
SET CLASSPATH=./classes;./lib/junit-4.5.jar
```

ЗАМЕЧАНИЕ: Вспомним, что символ точка означает текущую директорию .

При выполнении компиляции задается аргумент **-d**, значение которого **./classes** указывает на директорию, где должны размещаться файлы классов, полученные в результате компиляции. Обратите внимание на то, что символ разделителя директорий (/) может быть указан, как для платформы Windows, так и для платформы Unix. При компиляции множества классов из одного пакета можно использовать символ *:

```
javac -d ./classes ./src/zoostore/model/*.java
```

ЗАМЕЧАНИЕ: При выполнении название стартового класса должно быть полностью квалифицировано – с указанием цепочки иерархии пакетов.

ИТОГИ

В теме рассмотрены:

- Пакеты
- Компиляция и выполнение приложения с классами, размещенными в разных пакетах
- Переменная **CLASSPATH**