Java. Уровень 1

Урок 6

Продвинутое ООП

Углубленное изучение вопросов объектно-ориентированного программирования: наследование, полиморфизм

План занятия

1. Разбор домашнего задания
2. Разбираем возникшие вопросы по базовой части ООП
3. Что такое наследование и зачем нужно
4. Примеры наследования
5. Модификаторы доступа с учетом наследования
6. Абстрактные классы и методы
7. Ответы на вопросы

n очень похожих классов

```
public class Cat {
   private String name;
   private String color;
   private int age;

   public Cat(String name, String color, int age) {
      this.name = name;
      this.color = color;
      this.age = age;
   }

   public void info() {
      System.out.println(“name: “ + name + “
color: ” + color + “ age: “ + age);
   }
}
```

```
public class Dog {
   private String name;
   private String color;
   private int age;

   public Dog(String name, String color, int age) {
      this.name = name;
      this.color = color;
      this.age = age;
   }

   public void info() {
      System.out.println(“name: “ + name + “
color: ” + color + “ age: “ + age);
   }
}
```

а также классы: Cow, Horse, Duck, Chicken, …

Можно ли как-то сократить такой код?

Наследование

Создаем так называемый родительский класс (суперкласс) Animal

```
public class Animal {
   protected String name;
   protected String color;
   protected int age;

   public Animal(String name, String color, int age) {
      this.name = name;
      this.color = color;
      this.age = age;
   }

   public void info() {
      System.out.println(“name: “ + name + “ color: ” + color + “ age: “ + age);
   }
}
```

Пока код не намного короче…

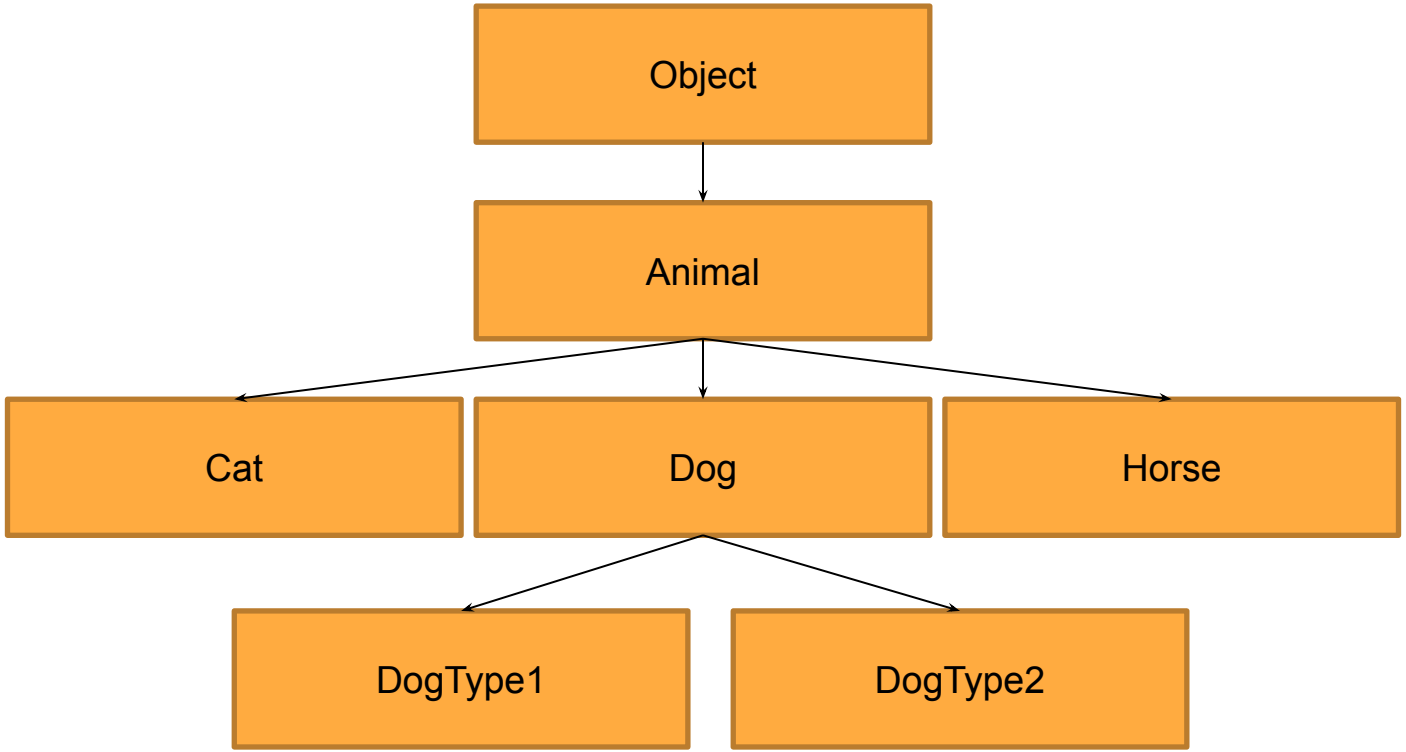
Посмотрим теперь на животных

```
public class Cat extends Animal {
   public Cat(String name, String color, int age) {
      super(name, color, age);
   }
}
 public class Dog extends Animal {
   public Dog(String name, String color, int age) {
      super(name, color, age);
   }
}
 public class Horse extends Animal {
   public Horse(String name, String color, int age) {
      super(name, color, age);
   }
}
```

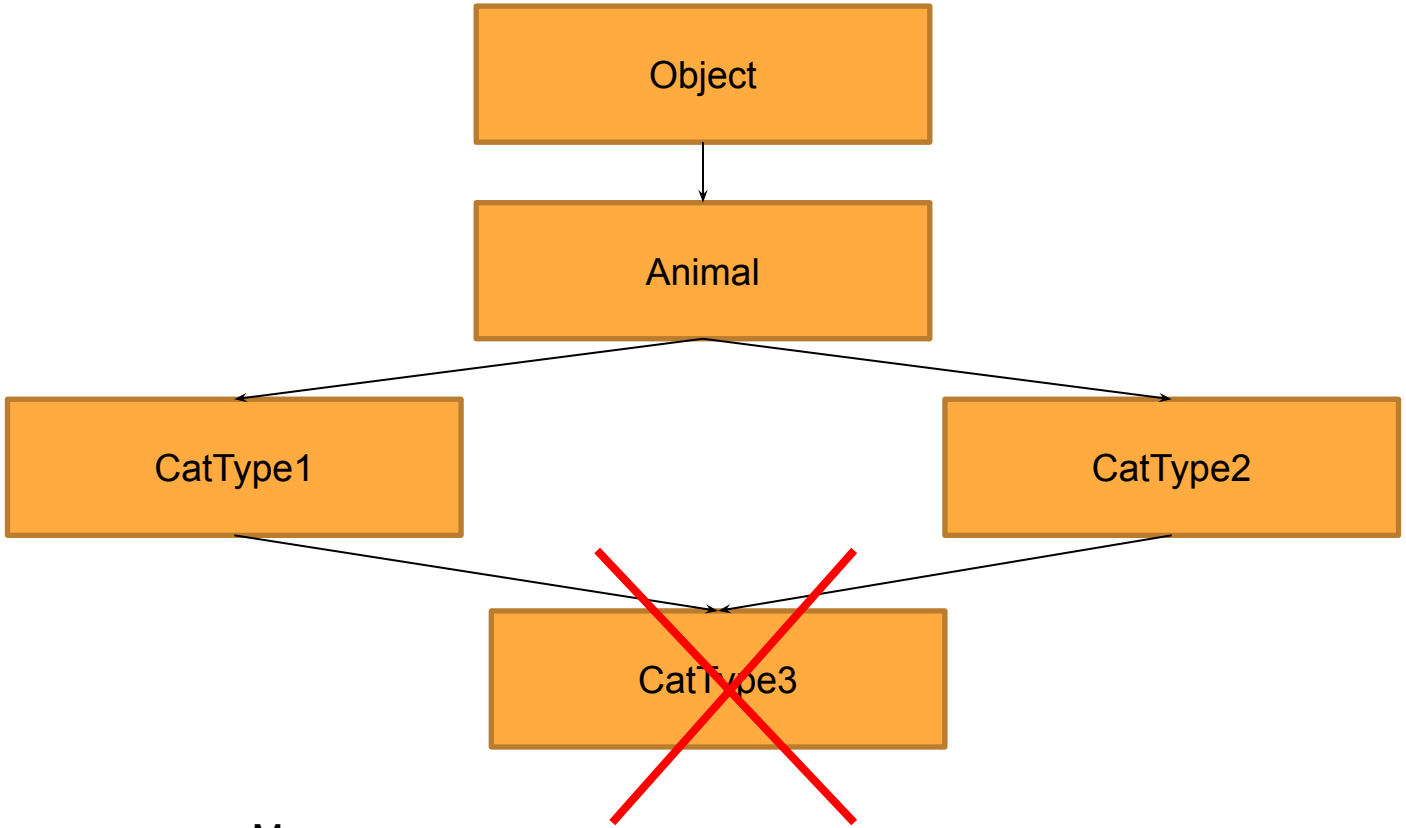
```
public class Duck extends Animal {
  public Duck(String name, String color, int age) {
     super(name, color, age);
  }
}
 public class Chicken extends Animal {
   public Chicken(String name, String color, int age) {
      super(name, color, age);
   }
}
 public class Pig extends Animal {
   public Pig(String name, String color, int age) {
      super(name, color, age);
   }
}
```

Довольно много классов уместилось на одной странице

Возможный вариант наследования



Недопустимый вариант наследования



Множественное наследование не поддерживается

Модификаторы доступа

	private	Модификатор отсутствует	protected	public
Один и тот же класс	+	+	+	+
Подкласс, производный от класса из того же самого пакета	-	+	+	+
Класс из того же самого пакета, не являющийся подклассом	-	+	+	+
Подкласс, производный от класса другого пакета	-	-	+	+
Класс из другого пакета, не являющийся подклассом, производный от класса данного пакета	-	-	-	+

Абстрактные классы и методы

```
public abstract class Animal {
    private String name;
    public abstract void voice();
    public void info() {
        System.out.println(“name: “ + name);
    }
}
```

Особенности:

- Нельзя создать экземпляр абстрактного класса;
- Абстрактные методы – методы не имеющие тела, каждый наследник должен переопределить такой метод; (например, метод voice() не имеет смысла прописывать у Animal, поскольку все наследники будут реализовывать его по-разному)
- В абстрактном классе могут быть обычные методы;
- В обычном классе не может быть абстрактных методов.

Домашнее задание

1. Создать классы Собака и Кот с наследованием от класса Животное.
2. Животные могут выполнять действия: бежать, плыть, перепрыгивать препятствие. В качестве параметра каждому методу передается величина, означающая или длину препятствия (для бега и плавания), или высоту (для прыжков).
3. У каждого животного есть ограничения на действия (бег: кот 200 м., собака 500 м.; прыжок: кот 2 м., собака 0.5 м.; плавание: кот не умеет плавать, собака 10 м.).
4. При попытке животного выполнить одно из этих действий оно должно сообщить результат в консоль. (Например, dog1.run(150); -> результат: run: true).
5. * Добавить животным разброс в ограничениях. То есть у одной собаки ограничение на бег может быть 400 м., у другой 600 м.

Вопросы участников