Java

Модификаторы доступа

Лекция #12

Пустовалова О.Г.
доцент. каф. мат.мод.
ИММиКН ЮФУ

Содержание

МОДИФИКАТОРЫ ДОСТУПА

Модификаторы доступа

1. Модификатор доступа **private** — ограничивает область действия классом.

2. Модификатор доступа **public** — не ограничивает область действия.

3. Модификатор доступа **protected** — ограничивает область действия пакетом и всеми подклассами.

4. Модификатор доступа *отсутствует (***default***)* — область действия ограничивается пакетом по умолчанию.

МОДИФИКАТОРЫ ДОСТУПА

PUBLIC

Модификаторы доступа. Public

- **Модификатор public** — класс, метод, конструктор, и т.д. объявленные как public могут быть доступны из любого другого класса.
- Поэтому поля, методы, объявленные внутри **public класса** могут быть доступны из любого класса.

- Тем не менее, если к public классу в другом пакете мы пытаемся получить доступ, то **public класс приходится импортировать**.

- Метод ***main()*** должен быть публичным (**public**). В противном случае, он не может быть вызван с помощью java-интерпретатора, чтобы запустить класс.

Модификаторы доступа. Public. Пример. Часть 1

```
public class Cat {

    public String name;
    public int age;
    public int weight;

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

    public Cat() {    }

    public void sayMeow() {
        System.out.println("Мяу!");
    }}
```

Так как, только один класс может иметь модификатор **public**, то классы определяем в разных файлах.

Модификаторы доступа. Public. Пример. Часть 2

```
public class Main {

    public static void main(String[] args) {

        Cat cat = new Cat();
         cat.name = "";
         cat.age = -1000;
         cat.weight = 0;

        System.out.println("Возраст кота равен "+cat.age);
        System.out.println("Вес кота равен "+cat.weight);
        cat.sayMeow();
    }
}
```


```
Возраст кота равен -1000
Вес кота равен 0
Мяу!
```

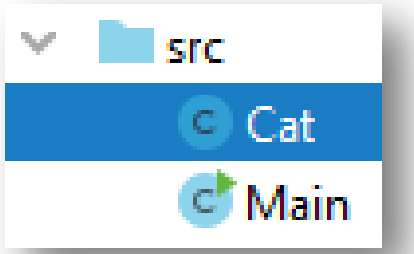
МОДИФИКАТОРЫ ДОСТУПА
PRIVATE

Модификаторы доступа. Private

- К переменной, методу или классу, помеченному модификатором **private**, можно обращаться только из того же класса, где он объявлен.
- Для всех остальных классов помеченный метод или переменная – невидимы.
- Это самая высокая степень закрытости – только свой класс.
- Такие методы не наследуются и не переопределяются.
- Доступ к ним из класса-наследника также невозможен.

Модификаторы доступа. Private. Пример. Часть 1

```
public class Cat {

    private String name;
    private int age;
    private int weight;

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

     public Cat() {    }
```

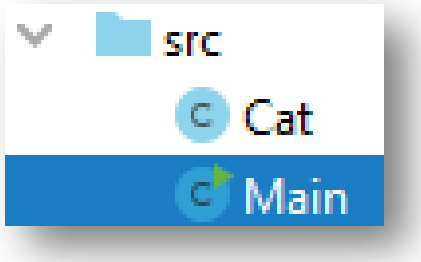


Модификаторы доступа. Private. Пример. Часть 2

```
    public void sayMeow() {  System.out.println("Мяу!"); }

    public String getName() {  return name; }

    public void setName(String name) { this.name = name; }

    public int getAge() { return age; }

    public void setAge(int age) { this.age = age; }

    public int getWeight() { return weight;}

    public void setWeight(int weight) { this.weight = weight; }
}
```

Модификаторы доступа. Private. Пример. Часть 3

```
public class Main {

    public static void main(String[] args) {

        Cat cat = new Cat("Tom",3,2500);

        System.out.println("Возраст кота равен "+cat.getAge());
        System.out.println("Вес кота равен "+cat.getWeight());
        cat.sayMeow();
    }
}
```




```
Возраст кота равен 3
Вес кота равен 2500
Мяу!
```

Модификаторы доступа. Private

- Ограничение доступа к полям и реализация геттеров-сеттеров — один из распространенных примеров использования **private** в реальной работе.

- Обычно, для реализации инкапсуляции в программе, используют этот модификатор.

МОДИФИКАТОРЫ ДОСТУПА
DEFAULT

Модификаторы доступа. Default

- Если переменная или метод не помечены никаким модификатором, то считается, что они помечены

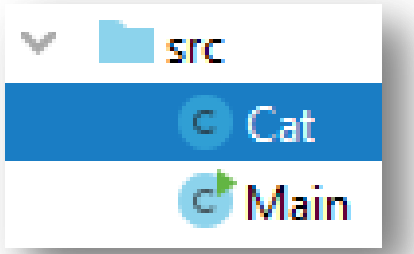
  модификатором по умолчанию

- Переменные и методы с таким модификатором видны всех классах пакета, в котором они объявлены, и только им.

- Этот модификатор еще называют «**package**» или «**package private**», намекая, что доступ к переменным и методам открыт для всего пакета, в котором находится их класс.

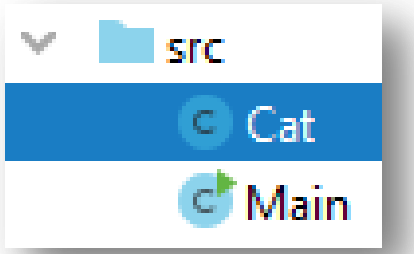
Модификаторы доступа. Default. Пример. Часть 1

```
public class Cat {

    String name;
    int age;
    int weight;

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

    Cat() {  }
```



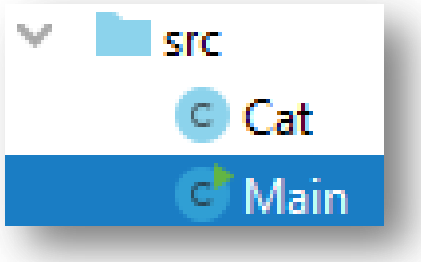
Модификаторы доступа. Default. Пример. Часть 2

```
void sayMeow() {  System.out.println("Мяу!"); }

String getName() {  return name; }

void setName(String name) { this.name = name; }

int getAge() { return age; }

void setAge(int age) { this.age = age; }

int getWeight() { return weight;}

void setWeight(int weight) { this.weight = weight; }
}
```



Модификаторы доступа. Default. Пример. Часть 3

```
public class Main {

    public static void main(String[] args) {

        Cat cat = new Cat("Tom",3,2500);

        System.out.println("Возраст кота равен "+cat.getAge());
        System.out.println("Вес кота равен "+cat.getWeight());
        cat.sayMeow();
    }
}
```




```
Возраст кота равен 3
Вес кота равен 2500
Мяу!
```

МОДИФИКАТОРЫ ДОСТУПА
PROTECTED

Модификаторы доступа. Protected

- Этот уровень доступа чуть шире, чем **package**.

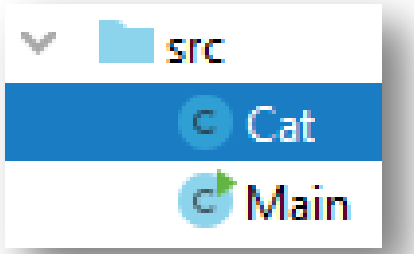
- К переменной, методу или классу, помеченному модификатором **protected**, можно обращаться из его же пакета (как package), но еще из всех классов, унаследованных от текущего.

Модификаторы доступа. Protected. Пример. Часть 1

```
public class Cat {

    protected String name;
    protected int age;
    protected int weight;

    protected Cat(String name, int age, int weight) {
        this.name = name;
        this.age = age;
        this.weight = weight;
    }

    protected Cat() {     }
```

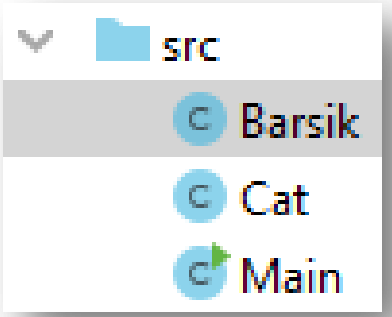
Модификаторы доступа. Protected. Пример. Часть 2

```
protected void sayMeow() {  System.out.println("Мяу!"); }

protected String getName() {  return name; }

protected void setName(String name) { this.name = name; }

protected int getAge() { return age; }

protected void setAge(int age) { this.age = age; }

protected int getWeight() { return weight;}

protected void setWeight(int weight) { this.weight = weight; }
```



Модификаторы доступа. Protected. Пример. Часть 3

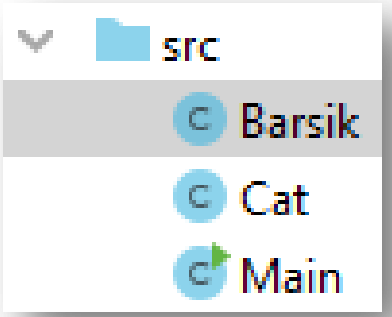
```
public class Barsik extends Cat {


    protected Barsik() {    }


    protected void sayMeowS() {
        System.out.println("Я Барсик!");
    }
}
```



Модификаторы доступа. Protected. Пример. Часть 4

```
public class Main {
    public static void main(String[] args) {
        Cat cat = new Cat();
        cat.sayMeow();

        Barsik bars = new Barsik();
        bars.setName("Барсик");
        bars.setAge(2);
        bars.setWeight(3000);

       System.out.println(bars.getName()+" "+bars.getAge()+" "+bars.getWeight());
        bars.sayMeowS();
        bars.sayMeow();
    }}
```

```
Мяу!
```




```
Барсик 2 3000
Я Барсик!
Мяу!
```

ЗАЩИЩЕННЫЙ ДОСТУП
PROTECTED

Защищенный доступ. protected

Как известно, поля в классе желательно объявлять как private, а некоторые методы — как public.

Любые закрытые (т.е. private) компоненты программы **невидимы** из других классов.

Это утверждение справедливо и для подклассов:

У подкласса отсутствует доступ к закрытым полям суперкласса

Но иногда приходится ограничивать доступ к некоторому методу и открывать его лишь для подклассов.

Реже возникает потребность предоставлять методам из подкласса доступ к полям в суперклассе.

В таком случае компонент класса объявляется защищенным с помощью модификатора доступа, обозначаемого ключевым словом **protected**.

Защищенный доступ. Защищенные поля

На практике пользоваться защищенными полями следует очень аккуратно.

Допустим, что созданный вами класс, в котором имеются защищенные поля, используется другими разработчиками.

Без вашего ведома другие могут создавать подклассы, производные от вашего класса, тем самым получая доступ к защищенным полям.

В таком случае вы уже не сможете изменить реализацию своего класса, не уведомив об этом других заинтересованных лиц.

Но это противоречит самому духу ООП, поощряющему инкапсуляцию данных.

ВЫВОДЫ

Модификаторы доступа

default - доступен внутри пакета.

private - доступен внутри класса

protected - доступен внутри класса и для всех подклассов.

public - доступен всем

Модификаторы доступа

default

Этот модификатор присваивается переменной, методу или классу в том случае, если явно его не указываем.

В этом случае переменная, метод или класс доступны всем классам внутри пакета.

Модификаторы доступа

private

Если мы используем модификатор доступа **private** для переменной, метода или конструктора, то они будут видны только внутри этого класса.

Классы и интерфейсы не могут иметь модификатора доступа **private**.

Доступ к переменным возможен только через **getters** и **setters**.

Модификаторы доступа

protected

Переменные, методы и конструкторы с модификатором доступа **protected** доступны в подклассах в других пакетах и в любых классах в том же пакете.

Модификаторы доступа

public

Переменные, методы, класс и т.д. с модификатором доступа **public** доустпны из любого класса.

Единственный ньюанс заключается в том, что если мы патемся получить доступ к публичному классу из другого пакета, то мы должны импортировать класс с помощью ключевого слова **import**.

Спасибо за внимание!