Лекции

Карта квестов (/quests)

Список лекций (/quests/lectures)

CS50 (/quests/QUEST HARVARD CS50)

Android (/quests/QUEST\_GOOGLE\_ANDROID)

C

(/me)

## Как работает множественный catch

Java Syntax (/quests/QUEST\_JAVA\_SYNTAX)
9 уровень (/quests/lectures/?quest=QUEST\_JAVA\_SYNTAX&level=9), 7 лекция (/quests/lectures/questsyntax.level09.lecture07)

ОТКРЫТА



- И ещё немного интересных уроков. Мне так нравится преподавать.
- Хочу рассказать тебе, как работает множественный catch. Все очень просто: при возникновении исключения в блоке try, выполнение программы передаётся на первый catch.
- Если тип, указанный внутри круглых скобок блока catch, совпадает с типом объекта-исключения, то начинается выполнение кода внутри блока {}. Иначе переходим к следующему catch. Там проверка повторяется.
- Если блоки catch закончились, а исключение так и не было перехвачено, то оно выбрасывается дальше, а текущий метод аварийно завершается.
- Ясно. Будет выполнен тот catch, тип в котором совпадает с типом исключения.
- Да, верно. Но в реальности все немного сложнее. Дело в том, что классы можно наследовать друг от друга. И если класс «Корова» унаследовать от класса «Животное», то объект типа «Корова» можно хранить не только в переменной типа «Корова», но и в переменной типа «Животное».
- И?
- Т.к. все исключения унаследованы от классов Exception или RuntimeException (который тоже унаследован от Exception), то их все можно перехватить командами catch (Exception e) или catch (RuntimeException e).
- -- И?
- Отсюда два вывода. Во-первых, с помощью команды catch(Exception e) можно перехватить любое исключение вообще. Во-вторых порядок блоков catch имеет значение.

## Примеры:

— Возникший при делении на 0 ArithmeticException будет перехвачен во втором catch.

Код

```
1 trv
 2 {
 3
       System.out.println("Before method1 calling.");
       int a = 1 / 0;
 4
 5
       System.out.println("After method1 calling. Never will be shown.");
 6 }
7 catch (NullPointerException e)
8 {
9
       System.out.println("Reference is null. Exception has been caught.");
10 }
11 catch (ArithmeticException e)
12 {
13
       System.out.println("Division by zero. Exception has been caught.");
14 }
15 catch (Exception e)
16 {
17
       System.out.println("Any other errors. Exception has been caught.");
18 }
```

— В примере ниже возникший ArithmeticException будет перехвачен в первом catch, т.к. классы всех исключений унаследованы от Exception. T.e. Exception захватывает любое исключение.

Код

```
1 try
2 {
       System.out.println("Before method1 calling.");
3
4
       int a = 1/0;
5
       System.out.println("After method1 calling. Never will be shown.");
6 }
7 catch (Exception e)
8
9
       System.out.println("Any other errors. Exception has been caught.");
10 }
11 catch (NullPointerException e)
12 {
       System.out.println("Reference is null. Exception has been caught.");
13
14 }
15 catch (ArithmeticException e)
16 {
17
       System.out.println("Divided by zero. Exception has been caught.");
18 }
```

— В примере ниже исключение ArithmeticException не будет перехвачено, а будет выброшено дальше в вызывающий метод.

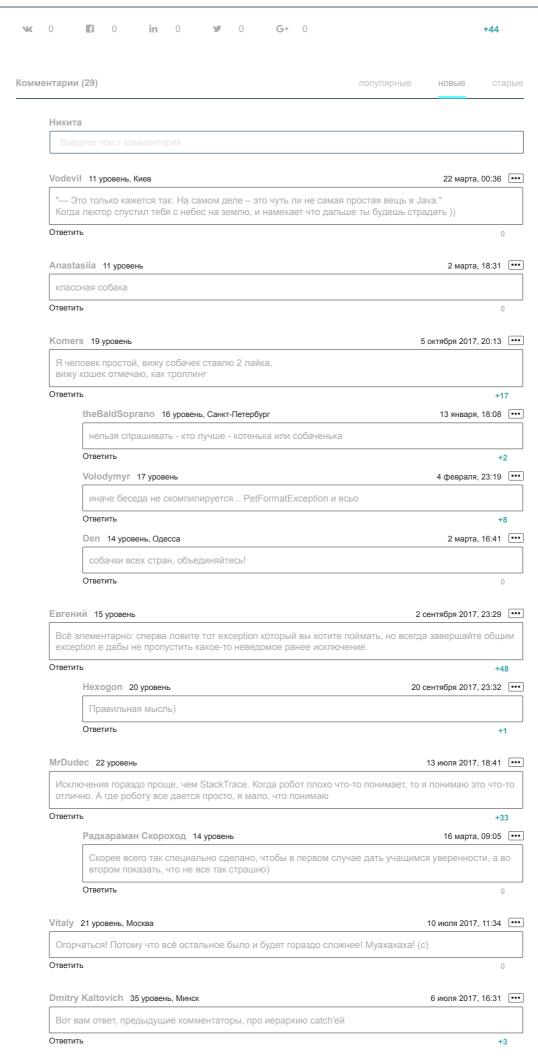
Код

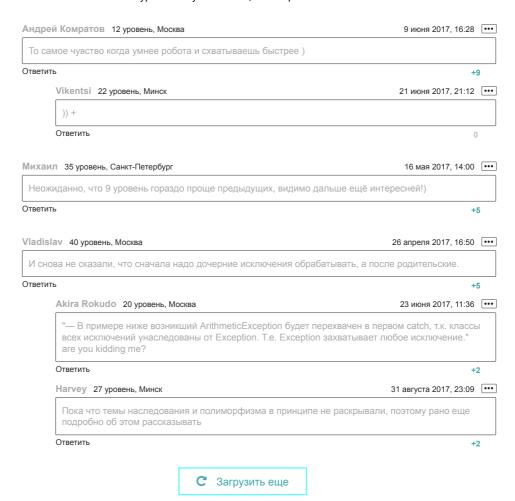
```
1 try
2 {
3    System.out.println("Before method1 calling.");
4    int a = 1/0;
5    System.out.println("After method1 calling. Never will be shown.");
6 }
7 catch (NullPointerException e)
8 {
9    System.out.println("Reference is null. Exception has been caught.");
10 }
```

- Ну, вроде все понемногу проясняется. Непростая штука эти исключения.
- Это только кажется так. На самом деле это чуть ли не самая простая вещь в Java.
- Не знаю, радоваться по этому поводу или огорчаться...

<(/quests/lectures/questsyntax.level09.lecture06)</pre>

×9 > (/quests/lectures/questsyntax.level09.lecture08)





<u>ush.ru/</u>) **G**+ (https://plus.google.com/114772402300089087607) **У** (https://twitter.com/javarush\_ru) (



Программистами не рождаются © 2018