



ОНЛАЙН-ОБРАЗОВАНИЕ

# 31 – Handling Exceptions (Часть 1)

**Дмитрий Коган**



## Как меня слышно и видно?



Если нет – напишите, если слышите – смайлик в чат.



## Цели :

- **Встретимся с магией**
- **Изучим классы исключений**
- **Будем держать всё под контролем**
- **Поймём преимущества исключений**





**Начинаем?**

# Темы экзамена

- ☐ Java Basics
- ☐ Working with Java Data Types
- ☐ Using Operators and Decision Constructs
- ☐ Creating and Using Arrays
- ☐ Using Loop Constructs
- ☐ Working with Methods and Encapsulation
- ☐ Working with Inheritance
- ☐ **Handling Exceptions**
- ☐ Working with Selected classes from the Java API

# Подтемы экзамена

## Handling Exceptions

- Differentiate among checked exceptions, unchecked exceptions, and Errors
- Create a try-catch block and determine how exceptions alter normal program flow
- Describe the advantages of Exception handling
- Create and invoke a method that throws an exception
- Recognize common exception classes (such as NullPointerException, ArithmeticException, ArrayIndexOutOfBoundsException, ClassCastException)



## Типы исключений



# Исключения

```
public class Zoo {  
    public static void main(String[] args) {  
        System.out.println(args[0]);  
        System.out.println(args[1]);  
    } }  

```

```
$ javac Zoo.java  
$ java Zoo Zoo
```

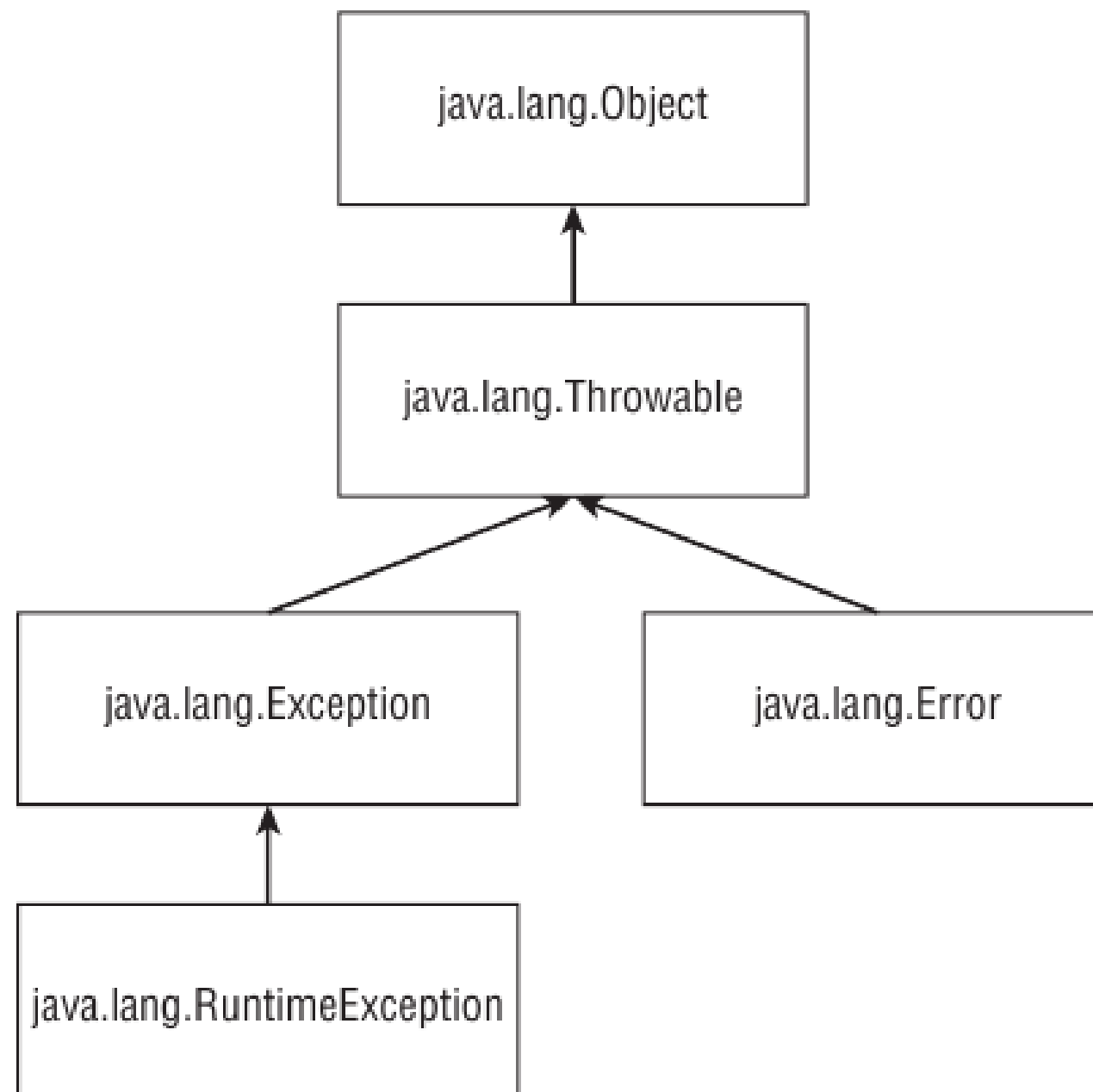
Zoo

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException:  
Index 1 out of bounds for length 1  
    at Zoo.main(Zoo.java:4)
```

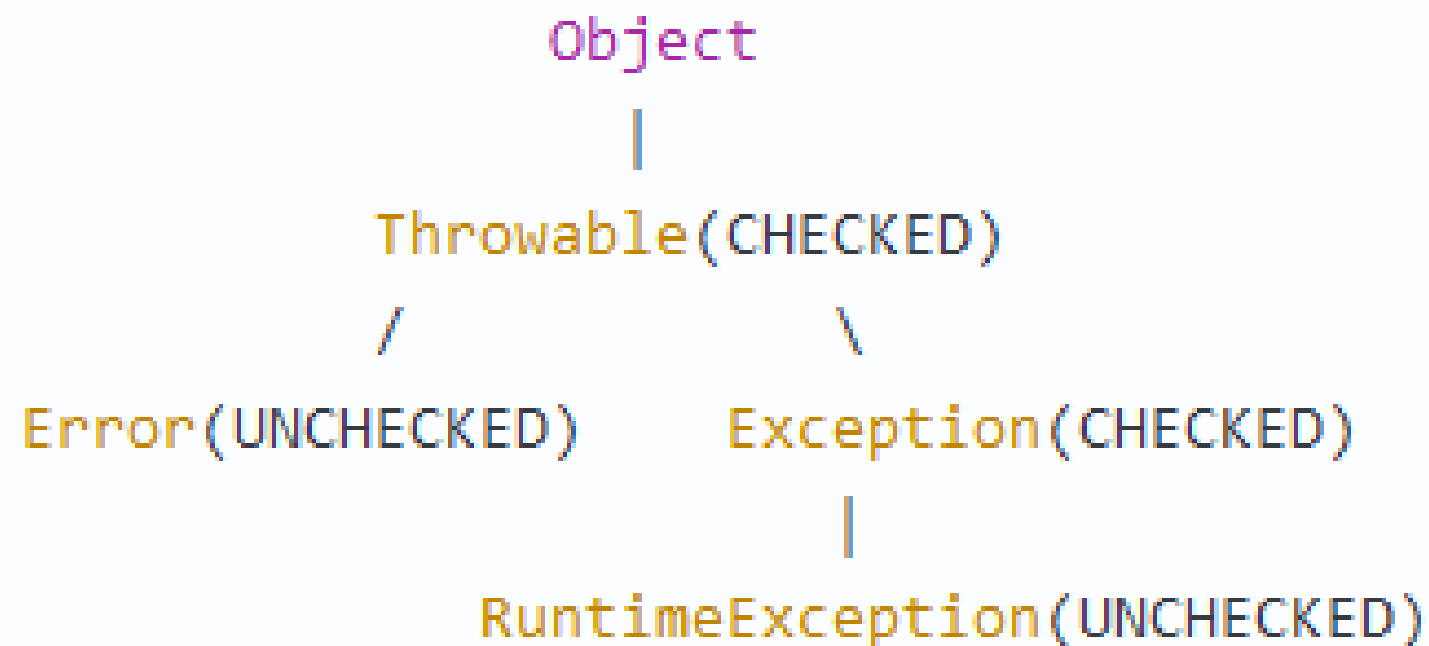
# Коды возврата

```
public int indexOf(String[] names, String name) {  
    for (int i = 0; i < names.length; i++) {  
        if (names[i].equals(name)) { return i; }  
    }  
    return -1;  
}
```

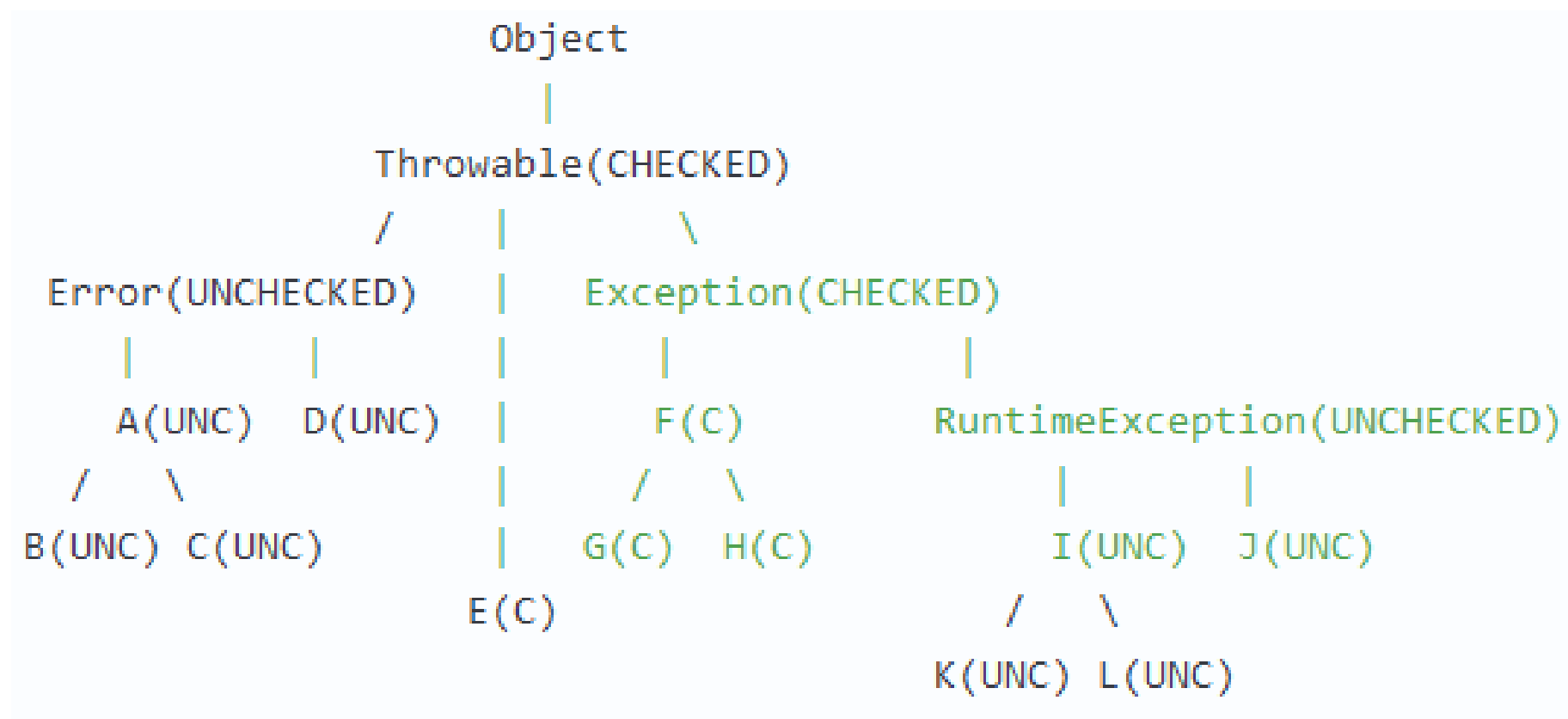
# Классы исключений



# Типы исключений



# Наследование ТИПОВ



# Сравнение типов

Type	How to recognize	Okay for program to catch?	Is program required to handle or declare?
Runtime exception	Subclass of RuntimeException	Yes	No
Checked exception	Subclass of Exception but not subclass of RuntimeException	Yes	Yes
Error	Subclass of Error	No	No



**Вопросы?**



## Магия исключений



# Возникновение

```
String[] animals = new String[0];  
System.out.println(animals[0]);
```

```
throw new Exception();  
throw new Exception("Ow! I fell.");  
throw new RuntimeException();  
throw new RuntimeException("Ow! I fell.");
```

# Исключение – объект

```
Exception e = new RuntimeException();  
throw e;
```

```
throw RuntimeException();    // DOES NOT COMPILE
```

# Недоступный код

```
try {  
    throw new RuntimeException();  
    throw new ArrayIndexOutOfBoundsException(); // DOES NOT COMPILE  
} catch (Exception e) {  
}
```

# Магия ключевых слов

```
public class App {  
    public static void main(String[] args) throws Throwable {}  
}
```

```
public class App {  
    public static void main(String[] args) throws String {}  
}
```

```
>> COMPILATION ERROR: Incompatible types: required 'java.lang.Throwable', found: 'java.lang.  
String'
```

# Магия ключевых слов

```
public class App {  
    public static void main(String[] args) {  
        try {  
        } catch (Throwable t) {}  
    }  
}
```

```
public class App {  
    public static void main(String[] args) {  
        try {  
        } catch (String s) {}  
    }  
}
```

```
>> COMPILATION ERROR: Incompatible types: required 'java.lang.Throwable', found: 'java.lang.  
String'
```

# Магия ключевых слов

```
public class App {  
    public static void main(String[] args) {  
        // Error - потому что Throwable  
        throw new Error();  
    }  
}
```

```
public class App {  
    public static void main(String[] args) {  
        throw new String("Hello!");  
    }  
}
```

```
>> COMPILATION ERROR: Incompatible types: required 'java.lang.Throwable', found: 'java.lang.  
String'
```

# null-аргумент

```
public class App {  
    public static void main(String[] args) {  
        throw null;  
    }  
}
```

```
>> RUNTIME ERROR: Exception in thread "main" java.lang.NullPointerException
```

# Анализируем

```
public class App {  
    public static void main(String[] args) {  
        f(null);  
    }  
    public static void f(NullPointerException e) {  
        try {  
            throw e;  
        } catch (NullPointerException npe) {  
            f(npe);  
        }  
    }  
}
```

```
>> RUNTIME ERROR: Exception in thread "main" java.lang.StackOverflowError
```



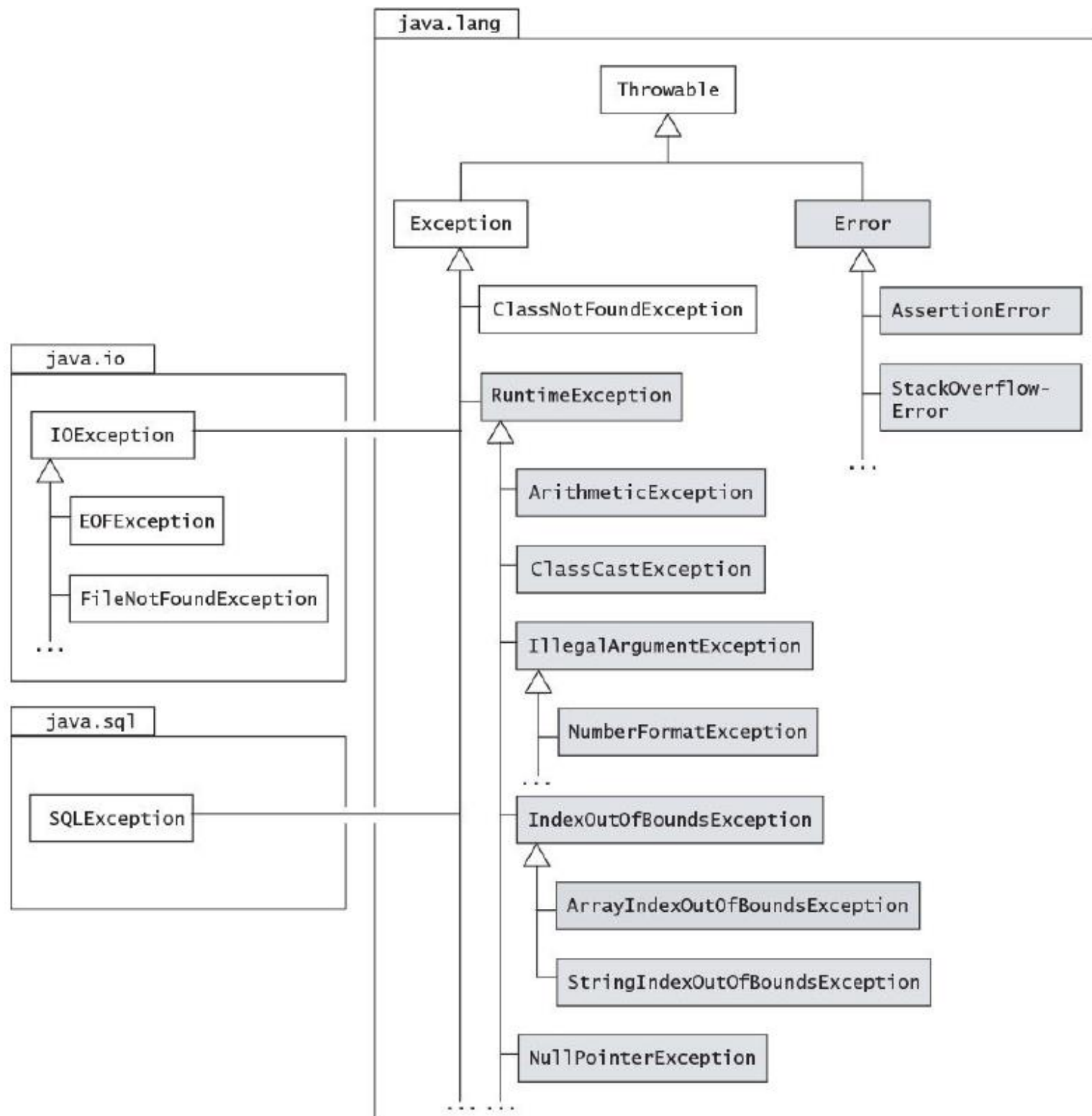


**Вопросы?**



## Классы исключений


# Классы и пакеты



Classes that are shaded (and their subclasses) represent unchecked exceptions.

# ArithmeticException

```
class ThrowArithmeticEx {  
    public static void main(String args[]) {  
        System.out.println(77/0);  
    }  
}
```



**Division  
by 0**

```
Exception in thread "main" java.lang.ArithmeticException: / by zero  
    at ThrowArithmeticEx.main(ThrowArithmeticEx.java:3)
```

```
System.out.println(0/0);
```



**Throws ArithmeticException**

# Infinity и NaN

```
class DivideDecimalNumberByZero {  
    public static void main(String args[]) {  
        System.out.println(77.0/0);  
    }  
}
```

← Outputs  
Infinity

```
class DivideNegativeDecimalNumberByZero {  
    public static void main(String args[]) {  
        System.out.println(-77.0/0);  
    }  
}
```

← Outputs  
-Infinity

```
System.out.println(0.0/0);
```

← Outputs NaN

```
class DivideIntegerByZeroPointZero {  
    public static void main(String args[]) {  
        System.out.println(77/0.0);  
        System.out.println(77.0/0.0);  
    }  
}
```

Outputs  
Infinity

# (A)IOOException

```
String[] season = {"Spring", "Summer"};  
ArrayList<String> exams = new ArrayList<>();  
exams.add("SCJP");  
exams.add("SCWCD");
```

```
System.out.println(season[5]);  
System.out.println(season[-9]);
```

Can't access position  
≥ array length

Can't access array at  
negative position

```
System.out.println(exams.get(-1));  
System.out.println(exams.get(4));
```

Can't access list at  
negative position

Can't access list at  
position ≥ its size

# ClassCastException

```
String type = "moose";  
Object obj = type;  
Integer number = (Integer) obj;
```

```
Exception in thread "main" java.lang.ClassCastException: java.lang.String  
cannot be cast to java.lang.Integer
```

# NullPointerException

**В каких случаях JVM бросает NullPointerException?**

1. Вызов instance-метода на null-объекте.
2. Попытка считать или изменить значение поля в null-объекте.
3. Обращение к полю length null-массива.
4. Попытка считать или изменить слот в null-массиве.
5. Попытка бросить null, словно это Throwable-объект.



# NullPointerException

```
String name;  
public void printLength() throws NullPointerException {  
    System.out.println(name.length());  
}
```

```
Exception in thread "main" java.lang.NullPointerException
```

# Что получим?

```
public class ExTest1
{
    public static void main(String[] args) throws Exception {
        int[] a = null;
        int i = a [ m() ];
    }
    public static int m() throws Exception {
        throw new Exception("Another Exception");
    }
}
```

# A ВОТ ЧТО!

```
public class ExTest1
{
    public static void main(String[] args) throws Exception {
        int[] a = null;
        int i = a [ m() ];
    }
    public static int m() throws Exception {
        throw new Exception("Another Exception");
    }
}
```

Exception in thread "main" java.lang.Exception: Another Exception

# IllegalArgumentException

```
public static void setNumberEggs(int numberEggs) {  
    if (numberEggs < 0)  
        throw new IllegalArgumentException(  
            "# eggs must not be negative");  
    this.numberEggs = numberEggs;  
}
```

Exception in thread "main" java.lang.IllegalArgumentException: # eggs must not be negative

# NumberFormatException

```
Integer.parseInt("abc");
```

```
Exception in thread "main" java.lang.NumberFormatException: For input string:  
"abc"
```

# NumberFormatException

```
public class ThrowNumberFormatException {  
    public static int convertToNum(String val) {  
        int num = 0;  
        try {  
            num = Integer.parseInt(val, 16);  
        } catch (NumberFormatException e) {  
            throw new NumberFormatException(val +  
                " cannot be converted to hexadecimal number");  
        }  
        return num;  
    }  
    public static void main(String args[]) {  
        System.out.println(convertToNum("16b"));  
        System.out.println(convertToNum("65v"));  
    }  
}
```

**In the exception handler,  
creates and throws new  
NumberFormatException  
with a custom message**

363

```
Exception in thread "main" java.lang.NumberFormatException: 65v cannot be  
converted to hexadecimal number  
    at ThrowNumberFormatException.convertToNum(ThrowNumberFormatException.java:8)  
    at ThrowNumberFormatException.main(ThrowNumberFormatException.java:14)
```

# Checked Exceptions

**IOException** Thrown programmatically when there's a problem reading or writing a file

**FileNotFoundException** Subclass of `IOException` thrown programmatically when code tries to reference a file that does not exist

# ExceptionInInitializerError

```
static {  
    int[] countsOfMoose = new int[3];  
    int num = countsOfMoose[-1];  
}  
public static void main(String[] args) { }
```

Exception in thread "main" java.lang.ExceptionInInitializerError  
Caused by: java.lang.ArrayIndexOutOfBoundsException: -1



# StackOverflowError

```
public static void doNotCodeThis(int num) {  
    doNotCodeThis(1);  
}
```

Exception in thread "main" java.lang.StackOverflowError

# NoClassDefFoundError

A `NoClassDefFoundError` occurs when Java can't find the class at runtime. Generally, this means a library available when the code was compiled is not available when the code is executed.

# К слову

- ✓ Запуск класса без правильного main – `NoSuchMethodError`.
- ✓ Если пихать в массив не то – `ArrayStoreException`.
- ✓ `SecurityException` – unchecked.

# Упражнение

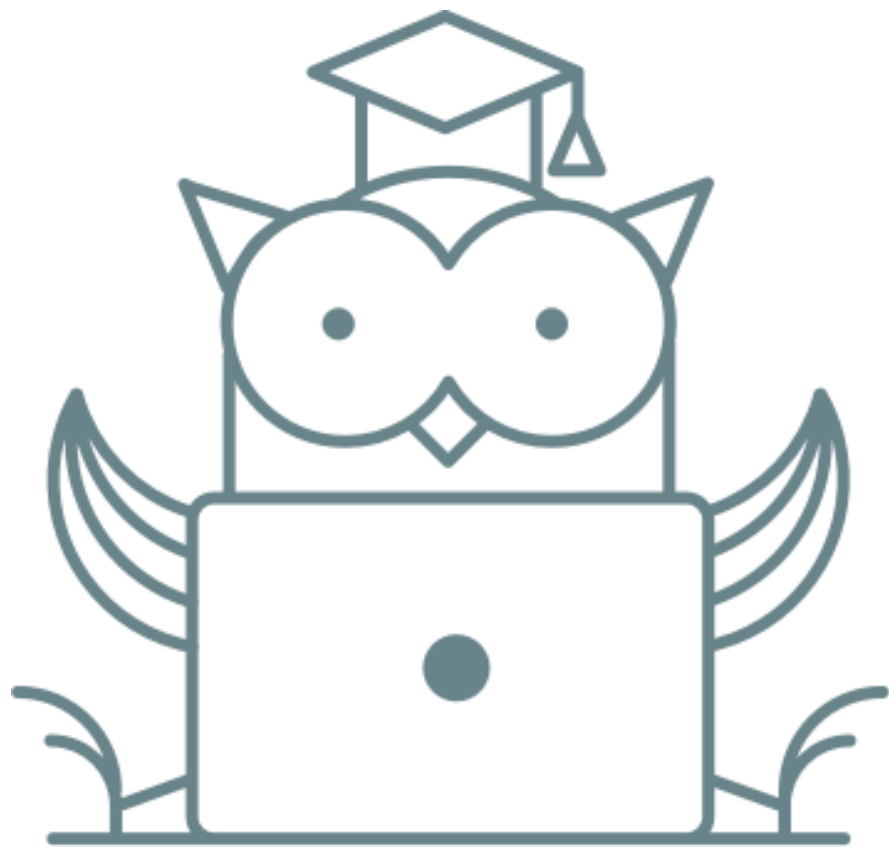
IndexOutOfBoundsException  
FileNotFoundException  
StackOverflowException  
IllegalFormatException  
ArrayOutOfBoundsException  
ClassCastException  
IOException  
IllegalArgumentException

**How many of them are defined in the java.lang package?**

- A. One
- B. Two
- C. Three
- D. Four
- E. Five
- F. Six
- G. Seven
- H. All of them



**Ответ: С**



**Вопросы?**



# Механизм возврата

# Сказано вернуть

```
public class App {  
    public double sqr(double arg) { // надо double  
        return arg * arg;           // double * double - это double  
    }  
}
```

```
public class App {  
    public double sqr(double arg) { // надо double  
        int k = 1;                 // есть int  
        return k;                  // можно неявно преобразовать int в double  
    }  
}
```

*// на самом деле, компилятор сгенерирует байт-код для следующих исходников*

```
public class App {  
    public double sqr(double arg) { // надо double  
        int k = 1;                 // есть int  
        return (double) k;         // явное преобразование int в double  
    }  
}
```



# Не выйdet

```
public class App {  
    public static double sqr(double arg) {  
        return "hello!";  
    }  
}
```

>> COMPILATION ERROR: Incompatible types. Required: `double`. Found: `java.lang.String`

```
public class App {  
    public static double sqr(double arg) {  
    }  
}
```

>> COMPILATION ERROR: Missing `return` statement

# И так не пройдет

```
public class App {  
    public static double sqr(double arg) {  
        if (System.currentTimeMillis() % 2 == 0) {  
            return arg * arg; // если currentTimeMillis() - четное число, то все OK  
        }  
        // а если нечетное, что нам возвращать?  
    }  
}
```

>> COMPILATION ERROR: Missing `return` statement

```
public class App {  
    public static void main(String[] args) {  
        double d = sqr(10.0); // ну, и чему равно d?  
        System.out.println(d);  
    }  
    public static double sqr(double arg) {  
        // nothing  
    }  
}
```

>> COMPILATION ERROR: Missing `return` statement

# Вешаем метод

```
public class App {  
    public static double sqr(double arg) {  
        while (true); // Удивительно, но КОМПИЛИРУЕТСЯ!  
    }  
}
```

```
public class App {  
    public static void main(String[] args) {  
        double d = sqr(10.0); // sqr - навсегда "повиснет", и  
        System.out.println(d); // d - НИКОГДА НИЧЕГО НЕ БУДЕТ ПРИСВОЕНО!  
    }  
    public static double sqr(double arg) {  
        while (true); // Вот тут мы на века "повисли"  
    }  
}
```

```
public class App {  
    public static double sqr(double arg) {  
        if (System.currentTimeMillis() % 2 == 0) {  
            return arg * arg; // ну ладно, вот твой double  
        } else {  
            while (true); // а тут "виснем" навсегда  
        }  
    }  
}
```

# Ничего не вернём

```
public class App {  
    public static double sqr(double arg) {  
        throw new RuntimeException();  
    }  
}
```

```
public class App {  
    public static double sqr(double arg) {// согласно объявлению метода ты должен вернуть do  
uble  
        long time = System.currentTimeMillis();  
        if (time % 2 == 0) {  
            return arg * arg; // ок, вот твой double  
        } else if (time % 2 == 1) { {  
            while (true); // не, я решил "повиснуть"  
        } else {  
            throw new RuntimeException(); // или бросить исключение  
        }  
    }  
}
```

# Никакого double

```
public class App {  
    public static void main(String[] args) {  
        // sqr - "сломается" (из него "выскочит" исключение),  
        double d = sqr(10.0); // выполнение метода main() прервется в этой строчке и  
                               // d - НИКОГДА НИЧЕГО НЕ БУДЕТ ПРИСВОЕНО!  
        System.out.println(d); // и печатать нам ничего не придется!  
    }  
    public static double sqr(double arg) {  
        throw new RuntimeException(); // "бросаем" исключение  
    }  
}
```

```
>> RUNTIME ERROR: Exception in thread "main" java.lang.RuntimeException
```

# Измеряем площадь

```
public static int area(int width, int height) {...}
```

```
public static int area(int width, int height) {  
    return width * height; // тут просто перемножаем  
}
```

```
public static int area(int width, int height) {  
    if (width < 0 || height < 0) {  
        // у вас плохие аргументы, извините  
    } else {  
        return width * height;  
    }  
}
```

```
>> COMPILATION ERROR: Missing return statement
```

# Измеряем площадь

```
public static int area(int width, int height) {  
    if (width < 0 || height < 0) {  
        System.out.println("Bad ...");  
    }  
    return width * height;  
}
```

```
public static int area(int width, int height) {  
    if (width < 0 || height < 0) {  
        return -1; // специальное "неправильное" значение площади  
    }  
    return width * height;  
}
```

```
public static int area(int width, int height) {  
    if (width < 0 || height < 0) {  
        System.exit(0);  
    }  
    return width * height;  
}
```

# Измеряем площадь

```
public static int area(int width, int height) {  
    if (width < 0 || height < 0) {  
        throw new IllegalArgumentException("Negative sizes: w = " + width + ", h = " + height);  
    }  
    return width * height;  
}
```



# Возврат исключения

```
static NullPointerException returnNPE() {  
    return new NullPointerException(); // Можно и throw new...,  
                                        // но так можно всегда (с любым return type)  
  
    //return ((String)null).length(); // Полетит NPE,  
                                        // но компилятор не согласен на такой return  
}  
  
static ArithmeticException returnAE() {  
    return new ArithmeticException();  
  
    // return 5/0; // этот фокус не проходит  
}
```



**Вопросы?**



# Контроль управления

# Передача управления

```
public class App {  
    public static void main(String[] args) {  
        // Пример: ОПЕРАТОР ПОСЛЕДОВАТЕЛЬНОСТИ  
        int x = 42;    // первый шаг  
        int y = x * x; // второй шаг  
        x = x * y;     // третий шаг  
        ...  
    }  
}
```

```
public class App {  
    public static void main(String[] args) {  
        // Пример: ОПЕРАТОР ВЕТВЛЕНИЯ  
        if (args.length > 2) { первый шаг  
            // второй шаг или тут  
            ...  
        } else {  
            // или тут  
            ...  
        }  
        // третий шаг  
        ...  
    }  
}
```

# return

```
public class App {
    public static void main(String[] args) {
        System.err.println("#1.in");
        f(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println("#1.out"); // вернулись
    } // выходим из текущего фрейма, кончились инструкции

    public static void f() {
        System.err.println(".    #2.in");
        g(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println(".    #2.out"); //вернулись
    } // выходим из текущего фрейма, кончились инструкции

    public static void g() {
        System.err.println(".    .    #3.in");
        h(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println(".    .    #3.out"); // вернулись
    } // выходим из текущего фрейма, кончились инструкции

    public static void h() {
        System.err.println(".    .    .    #4.in");
        if (true) {
            System.err.println(".    .    .    #4.RETURN");
            return; // выходим из текущего фрейма по 'return'
        }
        System.err.println(".    .    .    #4.out"); // ПРОПУСКАЕМ
    }
}

>> #1.in
>> .    #2.in
>> .    .    #3.in
>> .    .    .    #4.in
>> .    .    .    #4.RETURN
>> .    .    #3.out
>> .    #2.out
>> #1.out
```

# throw

```
public class App {
    public static void main(String[] args) {
        System.err.println("#1.in");
        f(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println("#1.out"); // ПРОПУСТИЛИ!
    }

    public static void f() {
        System.err.println(".  #2.in");
        g(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println(".  #2.out"); // ПРОПУСТИЛИ!
    }

    public static void g() {
        System.err.println(".  .  #3.in");
        h(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println(".  .  #3.out"); // ПРОПУСТИЛИ!
    }

    public static void h() {
        System.err.println(".  .  .  #4.in");
        if (true) {
            System.err.println(".  .  .  #4.THROW");
            throw new Error(); // выходим со всей пачки фреймов ("раскрутка стека") по 'throw'
        }
        System.err.println(".  .  .  #4.out"); // ПРОПУСТИЛИ!
    }
}

>> #1.in
>> .  #2.in
>> .  .  #3.in
>> .  .  .  #4.in
>> .  .  .  #4.THROW
>> RUNTIME ERROR: Exception in thread "main" java.lang.Error
```

# catch

```
public class App {
    public static void main(String[] args) {
        System.err.println("#1.in");
        try {
            f(); // создаем фрейм, помещаем в стек, передаем в него управление
        } catch (Error e) { // "перехватили" "летающее" исключение
            System.err.println("#1.CATCH"); // и работаем
        }
        System.err.println("#1.out"); // работаем дальше
    }

    public static void f() {
        System.err.println(". #2.in");
        g(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println(". #2.out"); // ПРОПУСТИЛИ!
    }

    public static void g() {
        System.err.println(". . #3.in");
        h(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println(". . #3.out"); // ПРОПУСТИЛИ!
    }

    public static void h() {
        System.err.println(". . . #4.in");
        if (true) {
            System.err.println(". . . #4.THROW");
            throw new Error(); // выходим со всей пачки фреймов ("раскрутка стека") по 'throw'
        }
        System.err.println(". . . #4.out"); // ПРОПУСТИЛИ!
    }
}

>> #1.in
>> . #2.in
>> . . #3.in
>> . . . #4.in
>> . . . #4.THROW
>> #1.CATCH
>> #1.out
```

# catch

```
public class App {
    public static void main(String[] args) {
        System.err.println("#1.in");
        f(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println("#1.out"); // вернулись и работаем
    }

    public static void f() {
        System.err.println(". #2.in");
        try {
            g(); // создаем фрейм, помещаем в стек, передаем в него управление
        } catch (Error e) { // "перехватили" "летающее" исключение
            System.err.println(". #2.CATCH"); // и работаем
        }
        System.err.println(". #2.out"); // работаем дальше
    }

    public static void g() {
        System.err.println(". . #3.in");
        h(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println(". . #3.out"); // ПРОПУСТИЛИ!
    }

    public static void h() {
        System.err.println(". . . #4.in");
        if (true) {
            System.err.println(". . . #4.THROW");
            throw new Error(); // выходим со всей пачки фреймов ("раскрутка стека") по 'throw'
        }
        System.err.println(". . . #4.out"); // ПРОПУСТИЛИ!
    }
}

>> #1.in
>> . #2.in
>> . . #3.in
>> . . . #4.in
>> . . . #4.THROW
>> . #2.CATCH
>> . #2.out
>> #1.out
```



# catch

```
public class App {
    public static void main(String[] args) {
        System.err.println("#1.in");
        f(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println("#1.out"); // вернулись и работаем
    }

    public static void f() {
        System.err.println(". #2.in");
        g(); // создаем фрейм, помещаем в стек, передаем в него управление
        System.err.println(". #2.out"); // вернулись и работаем
    }

    public static void g() {
        System.err.println(". . #3.in");
        try {
            h(); // создаем фрейм, помещаем в стек, передаем в него управление
        } catch (Error e) { // "перехватили" "летающее" исключение
            System.err.println(". . #3.CATCH"); // и работаем
        }
        System.err.println(". . #3.out"); // работаем дальше
    }

    public static void h() {
        System.err.println(". . . #4.in");
        if (true) {
            System.err.println(". . . #4.THROW");
            throw new Error(); // выходим со всей пачки фреймов ("раскрутка стека") по 'throw'
        }
        System.err.println(". . . #4.out"); // ПРОПУСТИЛИ!
    }
}

>> #1.in
>> . #2.in
>> . . #3.in
>> . . . #4.in
>> . . . #4.THROW
>> . . #3.CATCH
>> . . #3.out
>> . #2.out
>> #1.out
```

# Итого

```
// ---Используем RETURN--- // ---Используем THROW---
// Выход из 1-го фрейма // Выход из VCEX (из 4) фреймов
#1.in #1.in
. #2.in . #2.in
. . #3.in . . #3.in
. . . #4.in . . . #4.in
. . . #4.RETURN . . . #4.THROW
. . #3.out RUNTIME EXCEPTION: Exception in thread "main" java.lang.Error
. #2.out
#1.out
```

// ---Используем THROW+CATCH---		
// Выход из 3-х фреймов	// Выход из 2-х фреймов	// Выход из 1-го фрейма
#1.in	#1.in	#1.in
. #2.in	. #2.in	. #2.in
. . #3.in	. . #3.in	. . #3.in
. . . #4.in	. . . #4.in	. . . #4.in
. . . #4.THROW	. . . #4.THROW	. . . #4.THROW
#1.CATCH	. #2.CATCH	. . #3.CATCH
#1.out	. #2.out	. . #3.out
	#1.out	. #2.out
		#1.out



**Вопросы?**



# **Преимущества механизма исключений**

# Преимущества

- ✓ Обработка ошибок производится отдельно от «обычного» кода;
- ✓ Ошибки передаются вверх по стеку;
- ✓ Группирование и дифференциация ошибок согласно их типам.

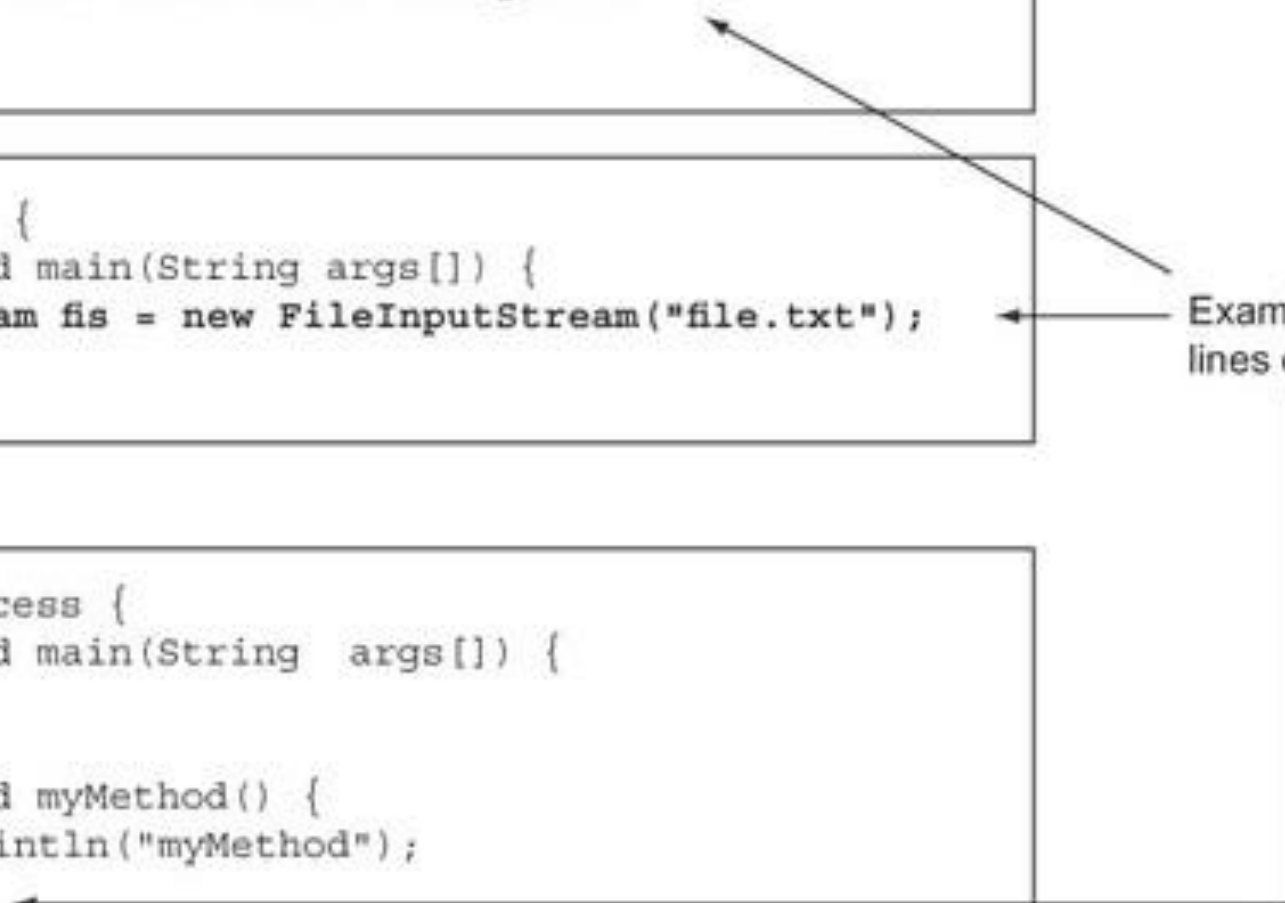
# Дифференциация

```
public class ArrayAccess {  
    public static void main(String args[]) {  
        String[] students = {"Shreya", "Joseph", null};  
        System.out.println(students[5].length());  
    }  
}
```

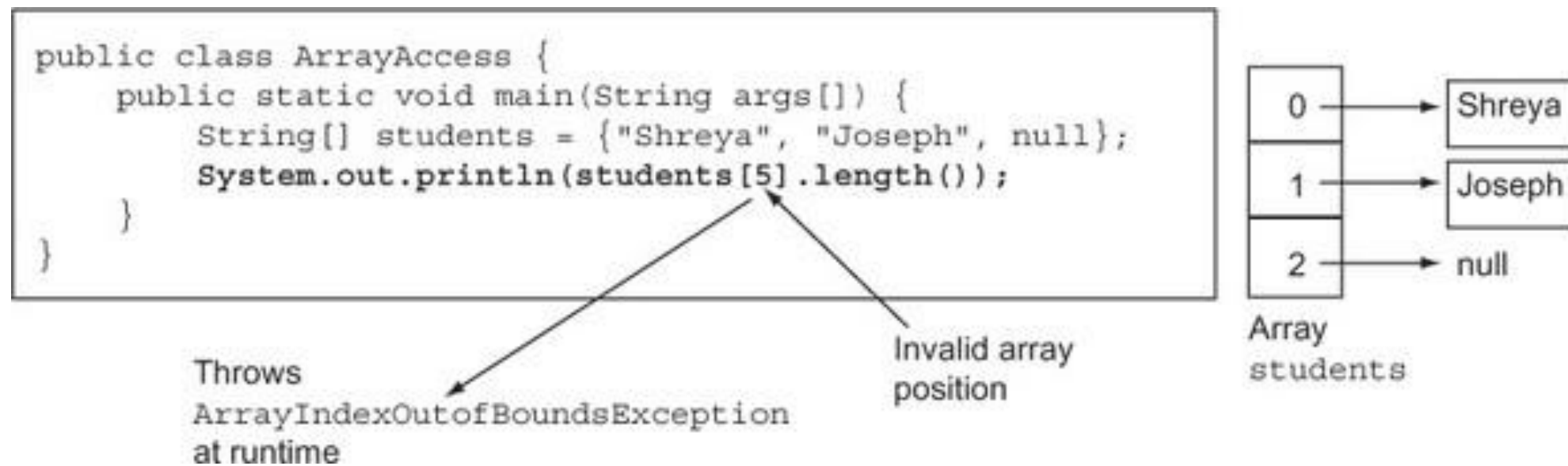
```
public class OpenFile {  
    public static void main(String args[]) {  
        FileInputStream fis = new FileInputStream("file.txt");  
    }  
}
```

```
public class MethodAccess {  
    public static void main(String args[]) {  
        myMethod();  
    }  
    public static void myMethod() {  
        System.out.println("myMethod");  
        myMethod();  
    }  
}
```

Examine these  
lines of code.



# Дифференциация



# Дифференциация

```
import java.io.*;
public class OpenFile {
    public static void main(String args[]) {
        FileInputStream fis = new
        FileInputStream("file.txt");
    }
}
```

Class fails  
to compile

Checked exception, FileNotFoundException, thrown by  
FileInputStream constructor, not "caught" by code



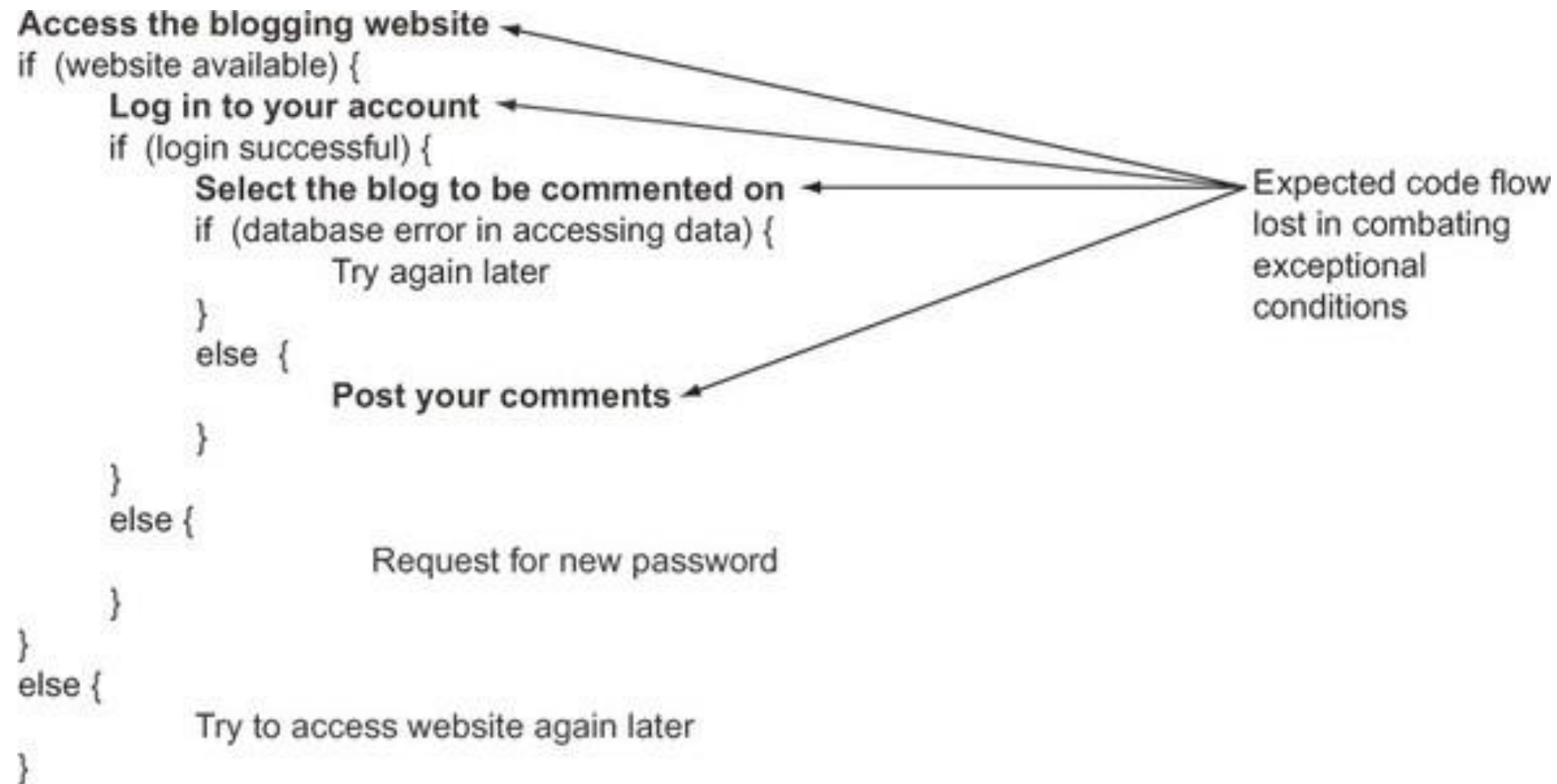
# Дифференциация

```
public class MethodAccess {  
    public static void main(String args[]) {  
        myMethod();  
    }  
    public static void myMethod() {  
        System.out.println("myMethod");  
        myMethod();  
    }  
}
```

Throws  
StackOverflowError at  
runtime

Calls itself recursively  
without an exit condition

# Отдельная обработка



# Отдельная обработка

```
try {  
    Access the blogging website  
    Log in to your account  
    Select the blog to be commented on  
    Post your comments
```

```
catch (WebsiteUnavailableException e) {  
    // define code to execute if website not available  
catch (LoginUnsuccessfulException e) {  
    // code to execute if login is unsuccessful  
catch (DatabaseAccessException e) {  
    // code to execute if data for particular  
    // post cannot be accessed  
}
```

Required code  
flows together.

Exception-handling  
code is separate  
from the regular  
flow of code.

# Вверх по стеку

```
public class Trace {                                // line 1
    public static void main(String args[]) {         // line 2
        method1();                                   // line 3
    }                                                 // line 4
    public static void method1() {                   // line 5
        method2();                                   // line 6
    }                                                 // line 7

    public static void method2() {                   // line 8
        String[] students = {"Shreya", "Joseph"};   // line 9
        System.out.println(students[5]);             // line 10
    }                                                 // line 11
}                                                    // line 12
```

Exception in thread "main"

java.lang.ArrayIndexOutOfBoundsException: 5

at Trace.method2(Trace.java:10)

at Trace.method1(Trace.java:6)

at Trace.main(Trace.java:3)

Offending code in method2 (line 10)

method2 called by method1 (line 6)

method1 called by main (line 3)



**Вопросы?**

---

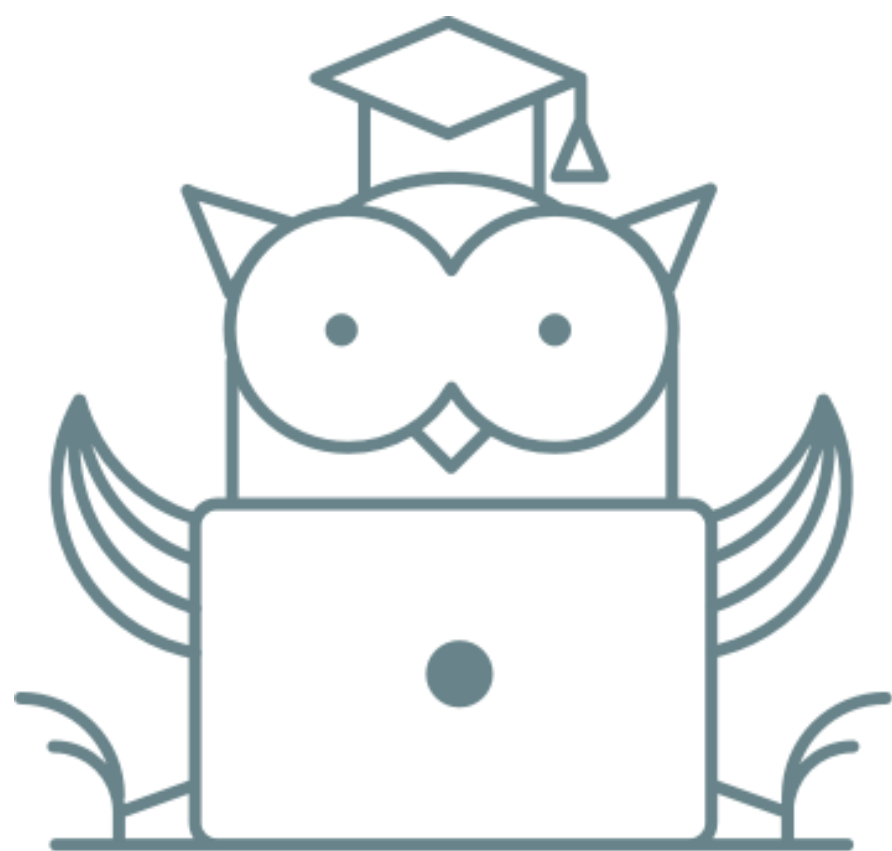
**Домашнее задание**

**Тест**



**Пожалуйста, пройдите опрос**

**<https://otus.ru/polls/17837/>**



**Спасибо  
за внимание!**

**Счастливых вам  
исключений!**