Java

Занятие 3. Исключения и обработка ошибок.

План занятия

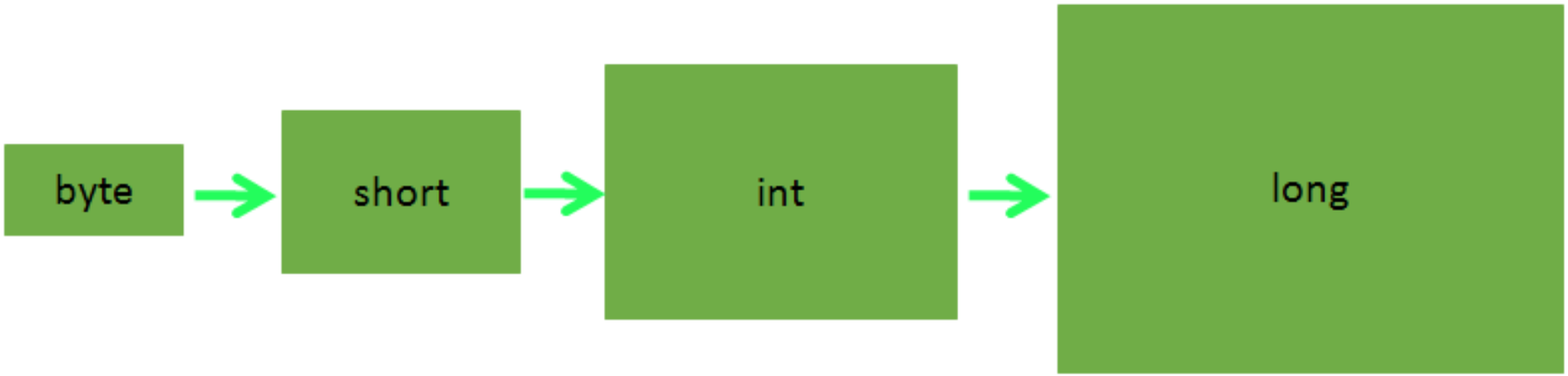
- Приведение типов
- Способы обработки ошибок
- Try-catch
- Виды исключений
- Finally
- Создание своих исключений
- Reflection

Приведение типов

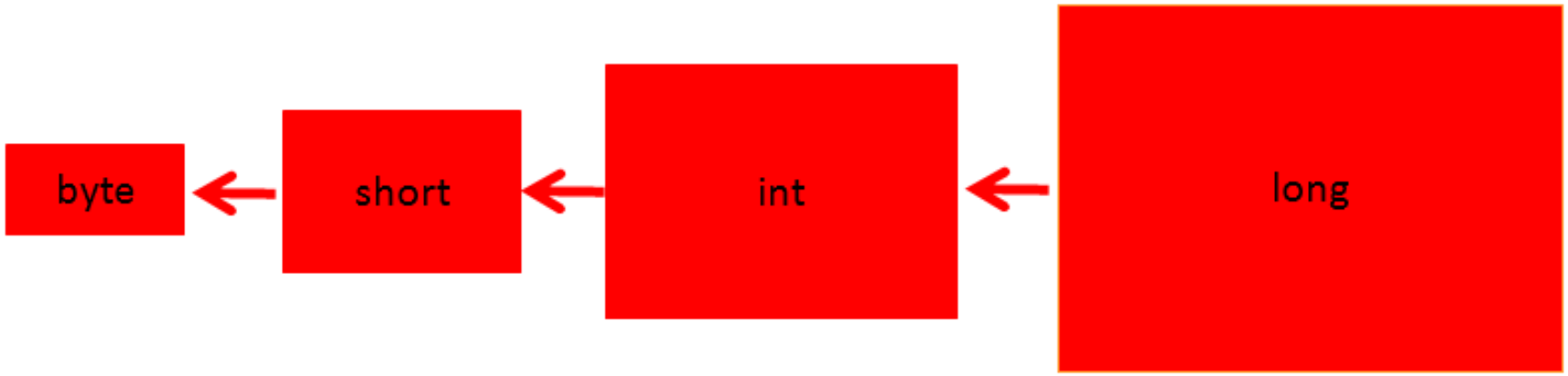
Ключевое слово	Тип	Диапазон допустимых значений (включительно)	Пример
byte	8-битное целочисленное значение	от -128 до 127	123
short	16-битное целочисленное значение	от -32768 до 32767	12345
int	32-битное целочисленное значение	от -2147483648 до 2147483647	1234567890
long	64-битное целочисленное значение	от -9223372036854775808 до 9223372036854775807	1234567890
float	32-битное значение с плавающей точкой	приблизительно ±3.40282347E+38F (6-7 значащих десятичных цифр)	123.45f
double	64-битное значение с плавающей точкой	приблизительно ±1.7976931348623157E+308F (15 значащих десятичных цифр)	123.456
char	16-битное значение Unicode	от 0 до 65535	‘a’
boolean	истина или ложь	true или false	true

Приведение типов

Автоматическое приведение типов



Явное приведение типов



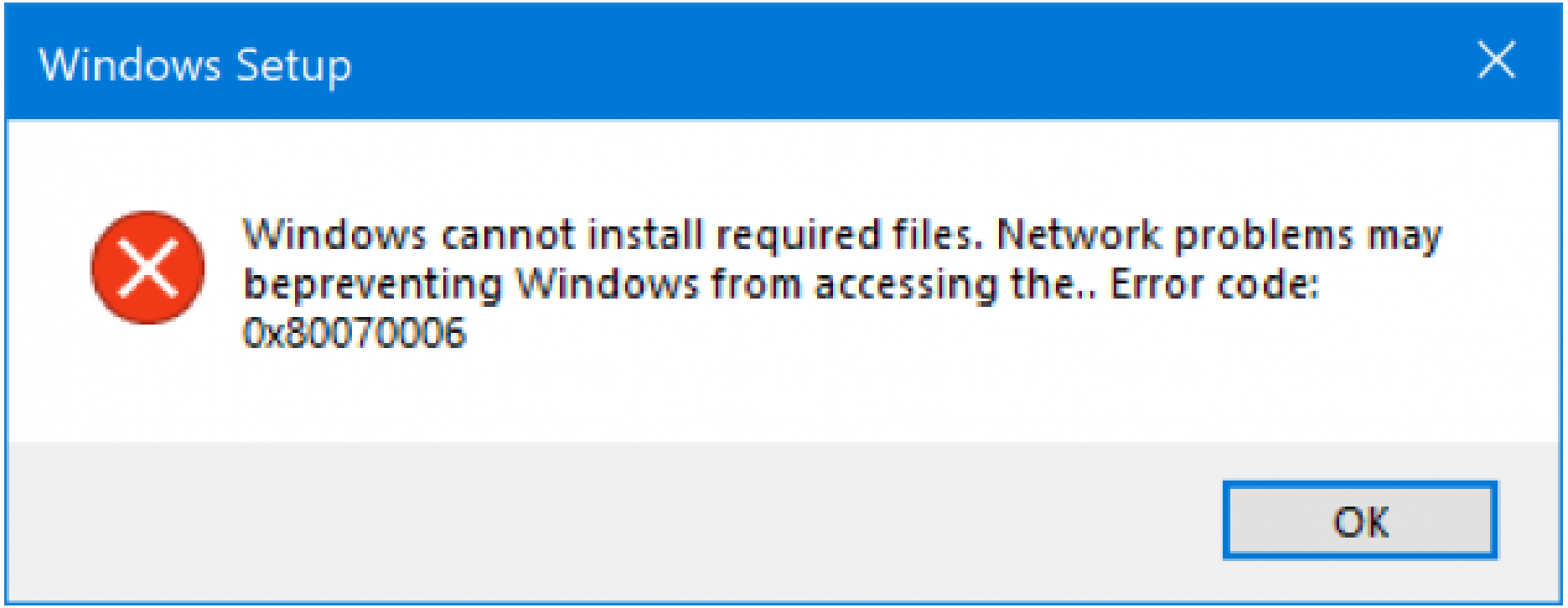
Полезные ссылки по приведению типов

https://javarush.ru/groups/posts/751-preobrazovanie-ssihlochnihkh-tipov-ili-spjajshiy-volk-na-klaviature

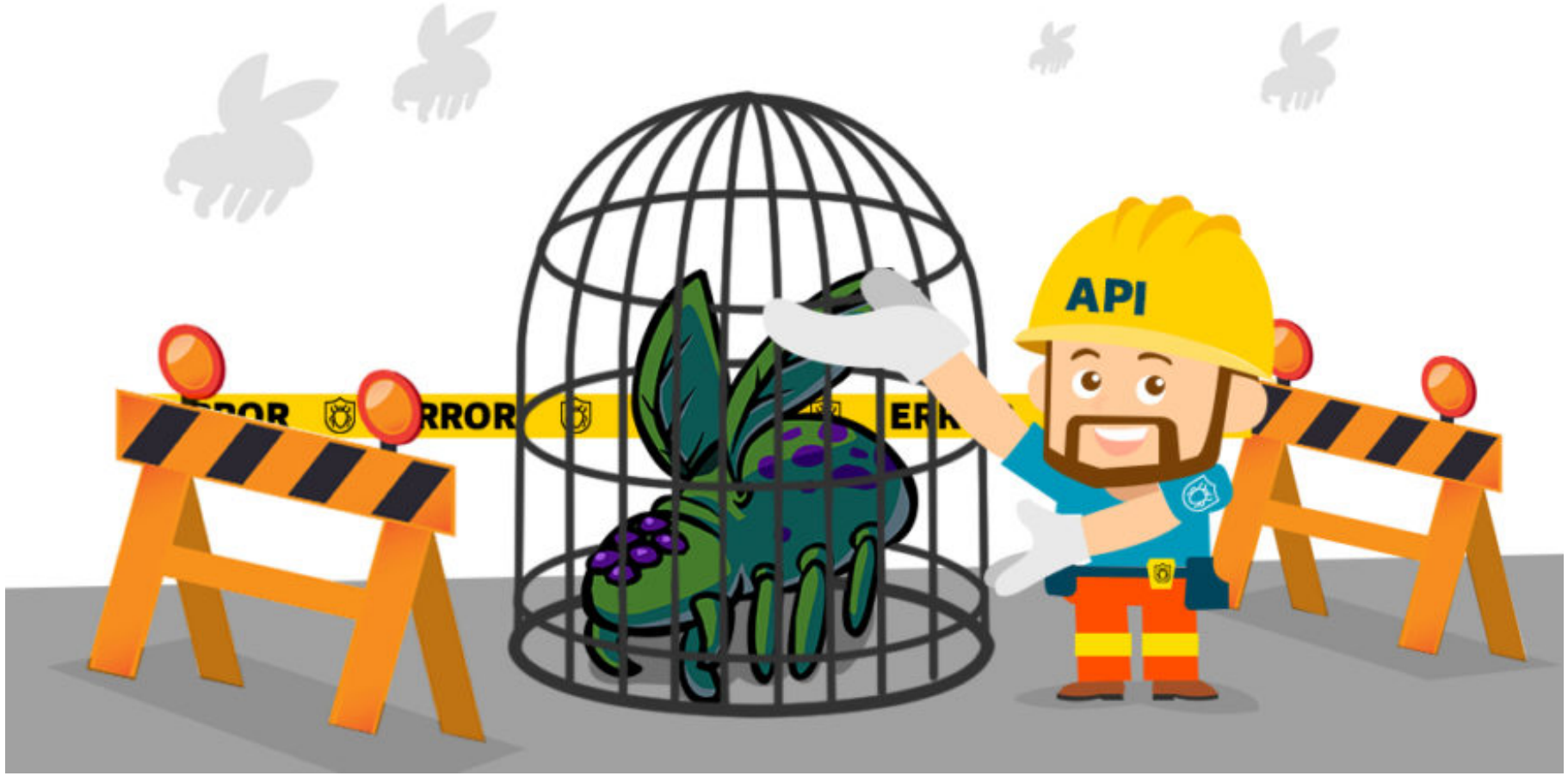
Способы обработки ошибок

Игнорировать ошибки

Хранить коды ошибок (C-style)



Механизм обработки исключений



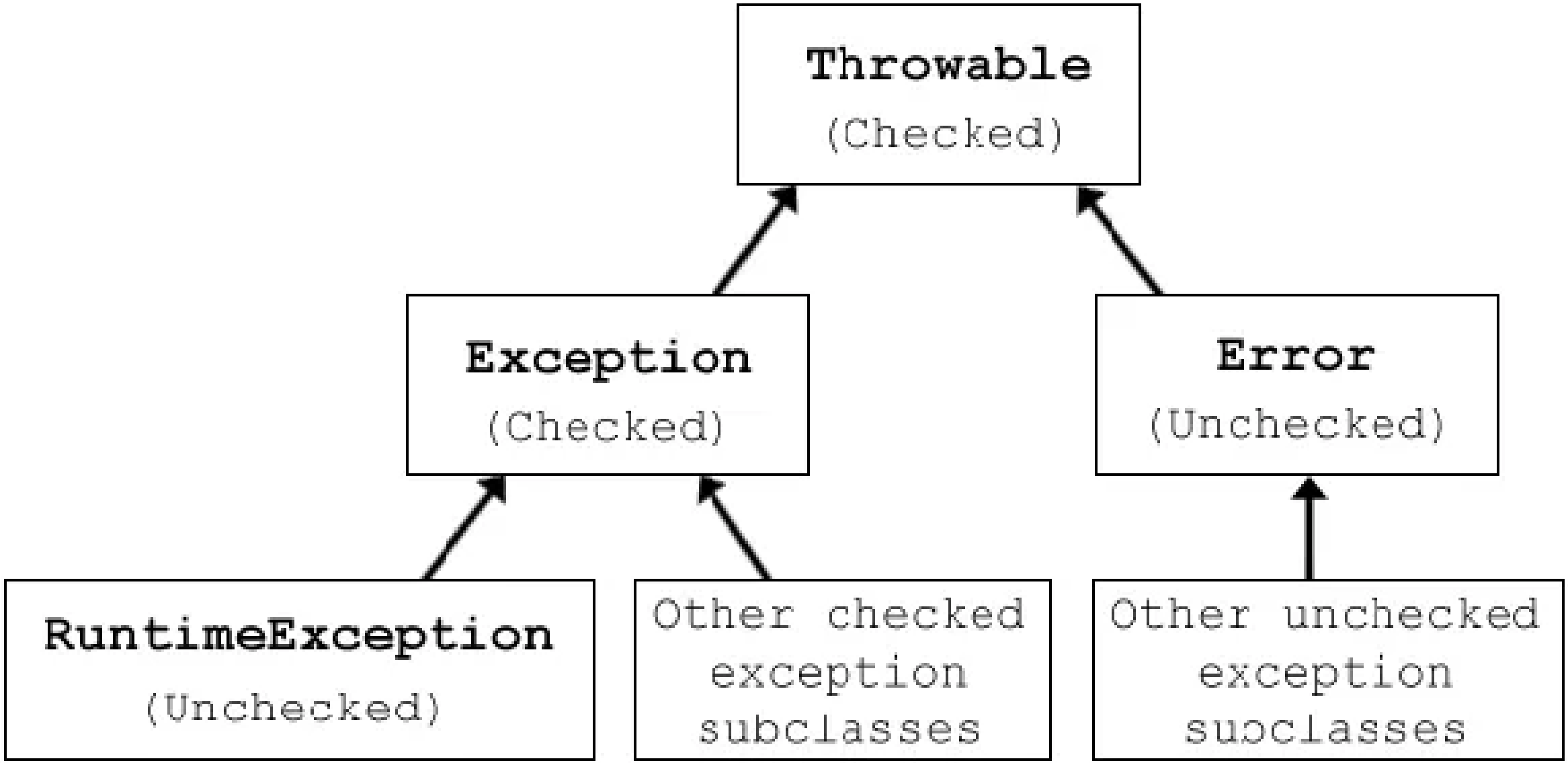
Исключение

Объект, хранит следующую информацию об ошибке:

- откуда
- стек вызовов
- сообщение (доп. информация)

Stack trace

```
Exception in thread "main" java.lang.Exception: this is exception in method
    at lesson3.ExceptionExample.exception(ExceptionExample.java:10)
    at lesson3.ExceptionExample.main(ExceptionExample.java:6)
```


Throwable
(Checked)
Exception
(Checked)
Error
(Unchecked)
RuntimeException
(Unchecked)
Other checked exception subclasses
Other unchecked exception subclasses

Типы ошибок

OutOfMemoryError

StackOverflowError

IllegalArgumentException

ArrayIndexOutOfBoundsException

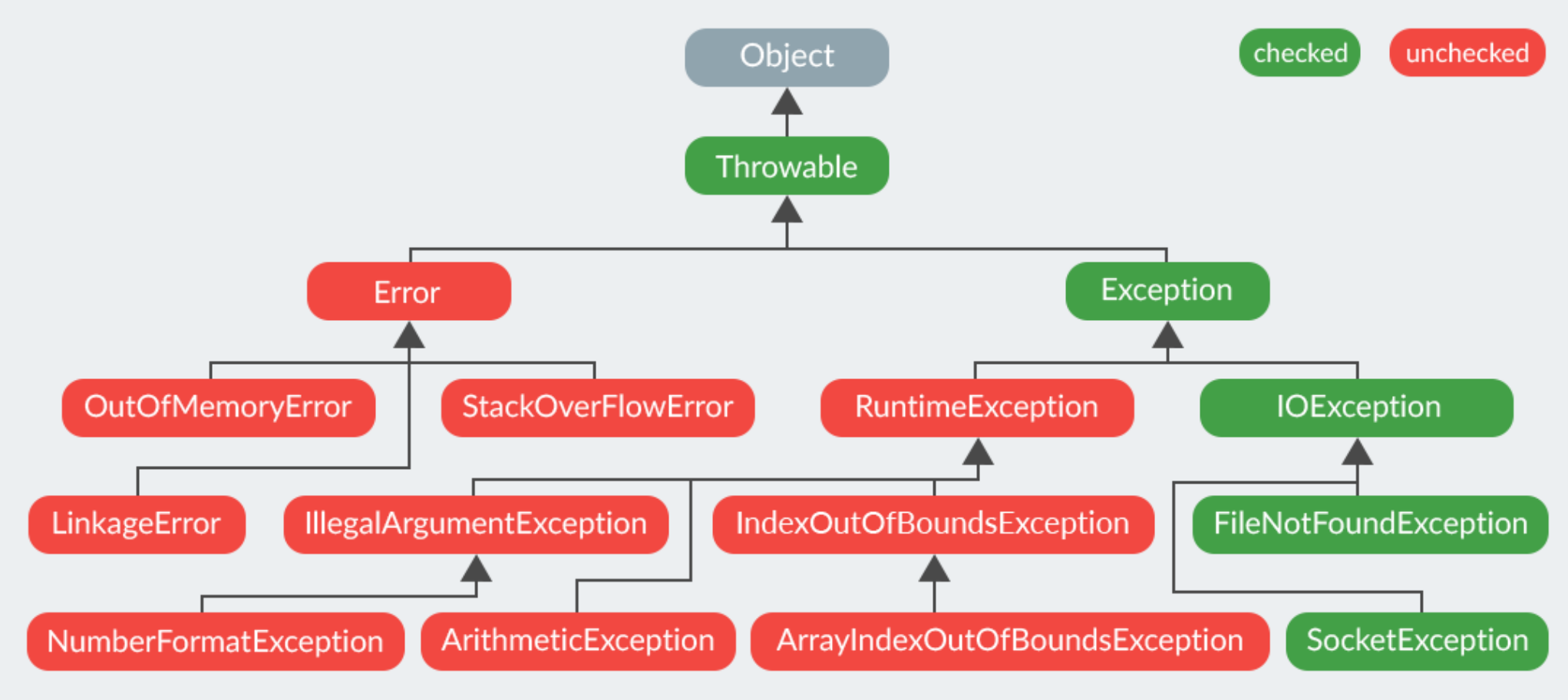
FileNotFoundException

IOException

ClassNotFoundException

NoSuchFieldException

Проверяемые и непроверяемые



Как не нужно делать

throw new NullPointerException(«invalid»);

(это ошибка разработчика)

try-catch

```
public static void main(String[] args) {
        String s = "i am not int";

        try {

            Integer integer = Integer.parseInt(s);

        } catch (NumberFormatException e) {

            e.printStackTrace();

        }
    }
```

Важно знать

try-catch — не транзакционный
(т. е. Все, что было вызвано до ошибки - исполнится)

Пробросить дальше

```
public static void cast(String s) throws NumberFormatException{
        Integer integer = Integer.parseInt(s);
    }
```

Свой exception

```
public class ComponentException extends Exception {

    public ComponentException(String message) {
        super(message);
    }

}
```

re-throw

```
public static Integer parse(String s) throws ComponentException {
        try {
            return Integer.parseInt(s);
        } catch (NumberFormatException e){
            throw new ComponentException("component exception");
        }
    }
```

Задача на исключения

main(): (хотим обработать IllegalArgExc)

m1() throws Exception
m2() throws IllegalArgExc
m3() throws IllegalArgExc, catch IOException
m4() throws IOException, IllegalArgExc
m5() throws IOException

Проблема

```
public static void open(File file){
    BufferedReader br;
    try {
        br = new BufferedReader(new FileReader(file));
        //do something useful
        br.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
}
```

finally - решение

```
public static void open(File file){
    BufferedReader br = null;
    try {
        br = new BufferedReader(new FileReader(file));
        //do something useful
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } finally {
        br.close();
    }
}
```

Ho...

```
public static void open(File file){
    BufferedReader br = null;
    try {
        br = new BufferedReader(new FileReader(file));
        //do something useful
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } finally {
        try {
            br.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Решение — try-with-resources

```
public static void open(File file) throws IOException {
        try (BufferedReader br = new BufferedReader(new FileReader(file))){
            //do something
        }
    }
```

Полезные ссылки по исключениям

https://habr.com/ru/post/178405/

https://habr.com/ru/post/183322/ (комментарии)

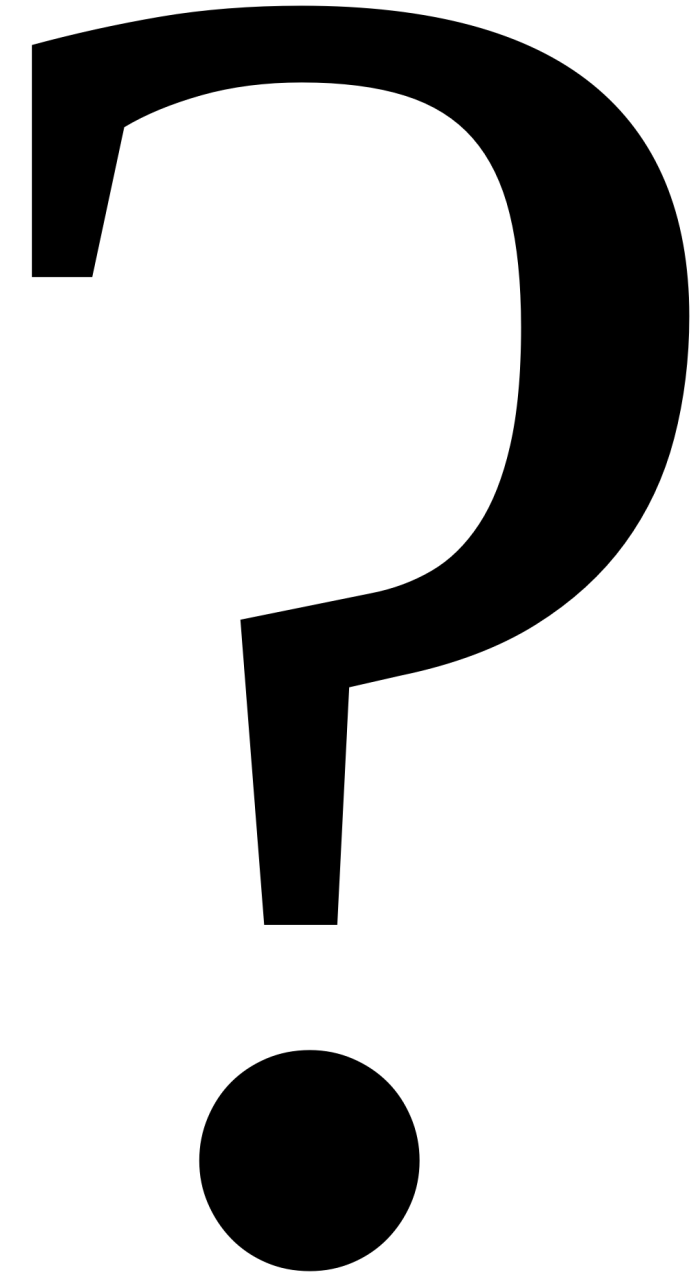
Reflection

Reflection

Находится в пакете java.lang

Методы java.lang.reflect позволяют узнать:

- Класс объекта;
- Получить информацию о модификаторах класса, полях, методах, константах, конструкторах и суперклассах;
- Выяснить, какие методы принадлежат реализуемому интерфейсу/интерфейсам;
- Создать экземпляр класса, причем имя класса неизвестно до момента выполнения программы;
- Получить и установить значение поля объекта по имени;
- Вызвать метод объекта по имени.




RESPONSE
OPINION
SURVEY
FEEDBACK
COMMENT
IDEA
ADVICE
RESULT
RATING