Java

Исключения и обработка ошибок

План занятия

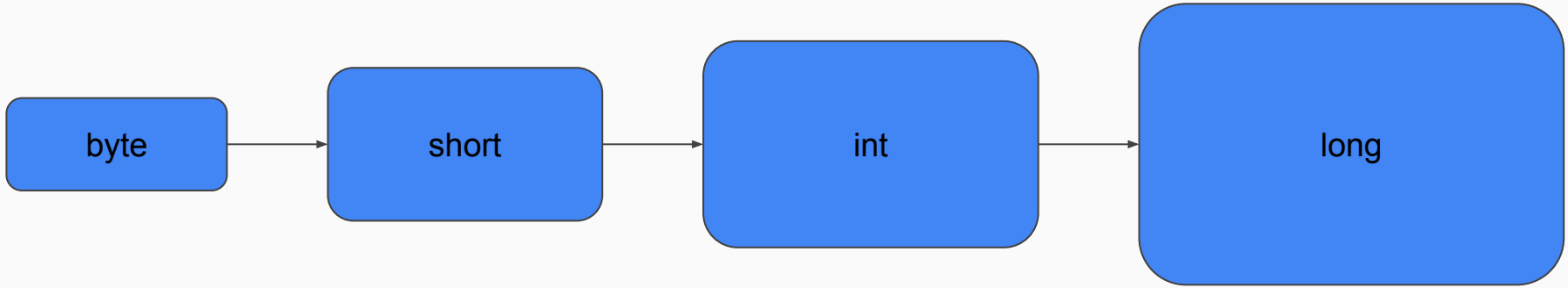
- Приведение типов
- Способы обработки ошибок
- try-catch
- Виды исключений
- finally
- Создание своих исключений
- Reflection

Домашнее задание 1

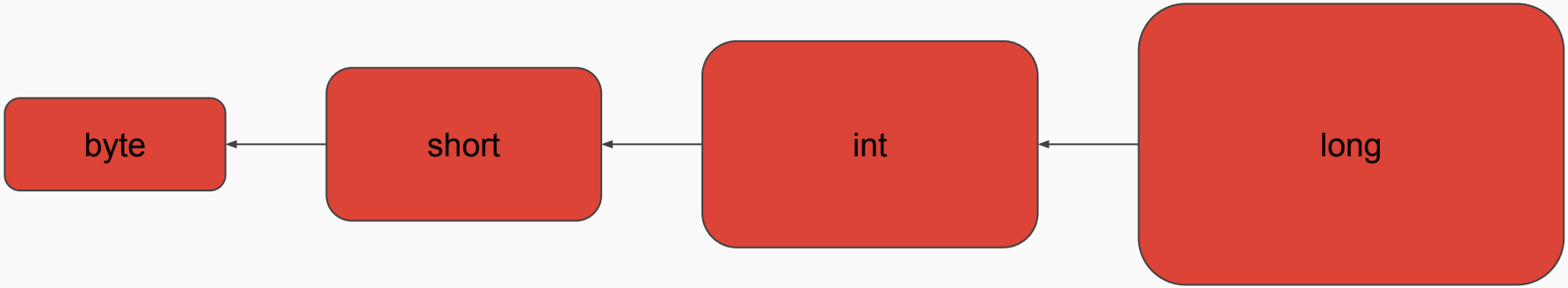
- Форматтер
- Давать переменным осмысленные имена
- Сравнение на equals лучше делать слева
- Осмысленные сообщения коммита
- Add files via upload
- Стоит учитывать тему занятия

Приведение типов

Автоматическое (неявное) приведение типов



Явное приведение типов



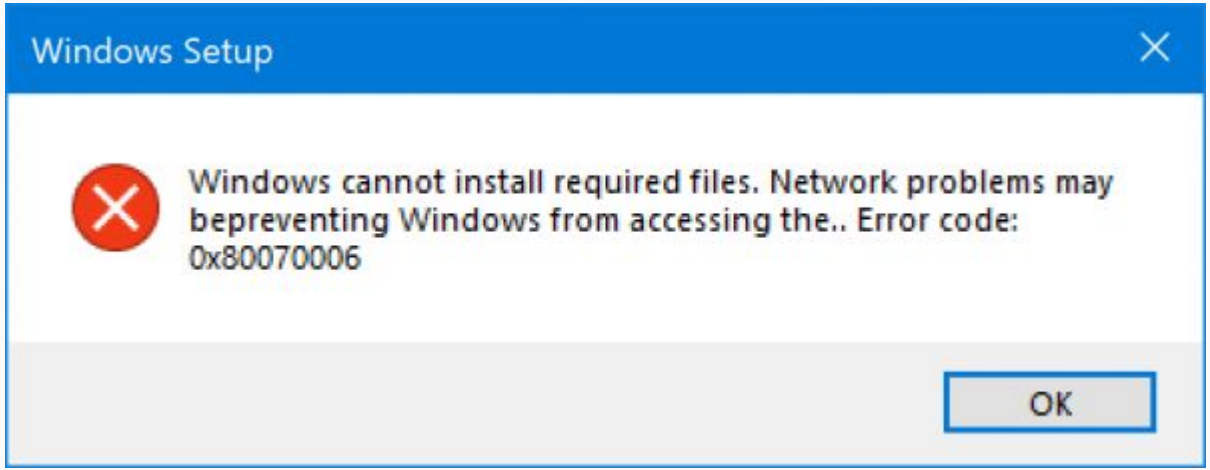
Приведение ссылочных типов

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

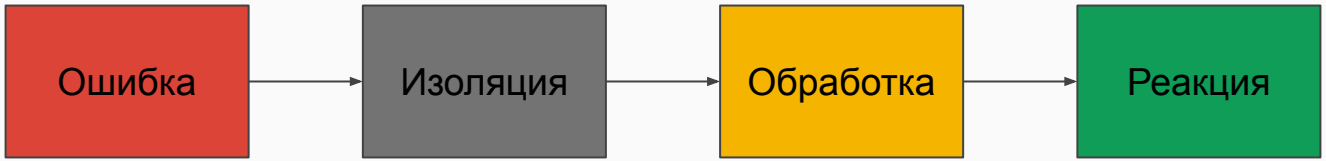
Способы обработки ошибок

- Игнорировать
- Обрабатывать

Обработка кодов ошибок



Обработка исключений



Исключение (Exception)

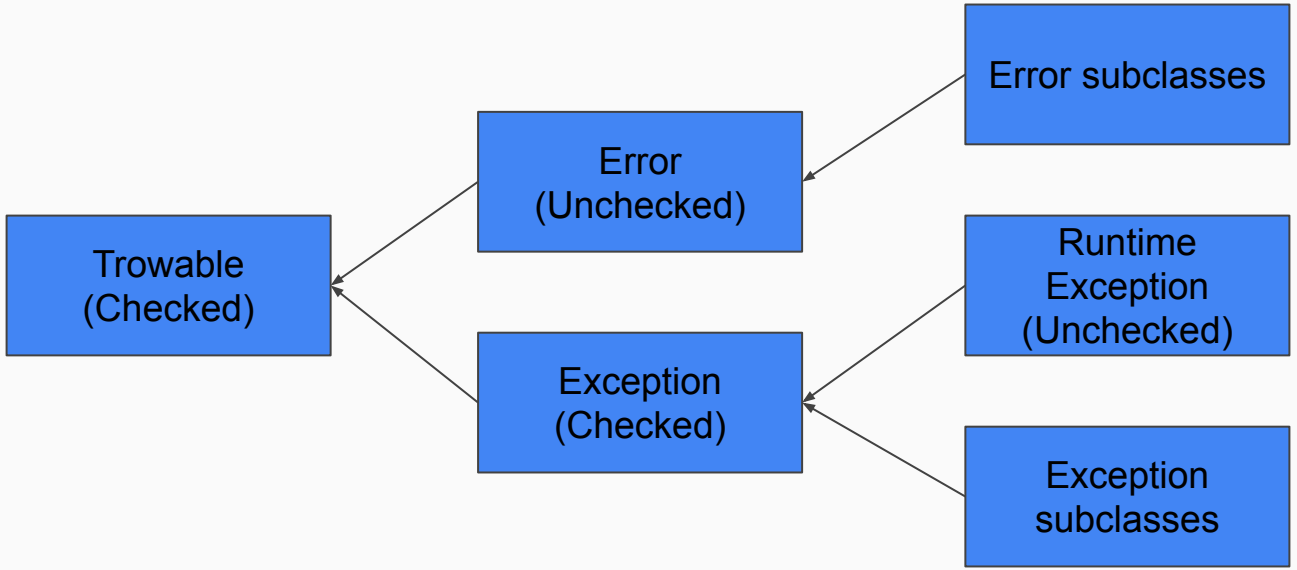
Исключение:

- Место
- Стек вызова
- Сообщение

Stack trace

```
Exception in thread "main" com.example.task3.MyOwnException Create breakpoint : My Message
    at com.example.task3.TestClassWithMain.testMethod3(TestClassWithMain.java:19)
    at com.example.task3.TestClassWithMain.testMethod2(TestClassWithMain.java:15)
    at com.example.task3.TestClassWithMain.testMethod1(TestClassWithMain.java:11)
    at com.example.task3.TestClassWithMain.main(TestClassWithMain.java:7)
```

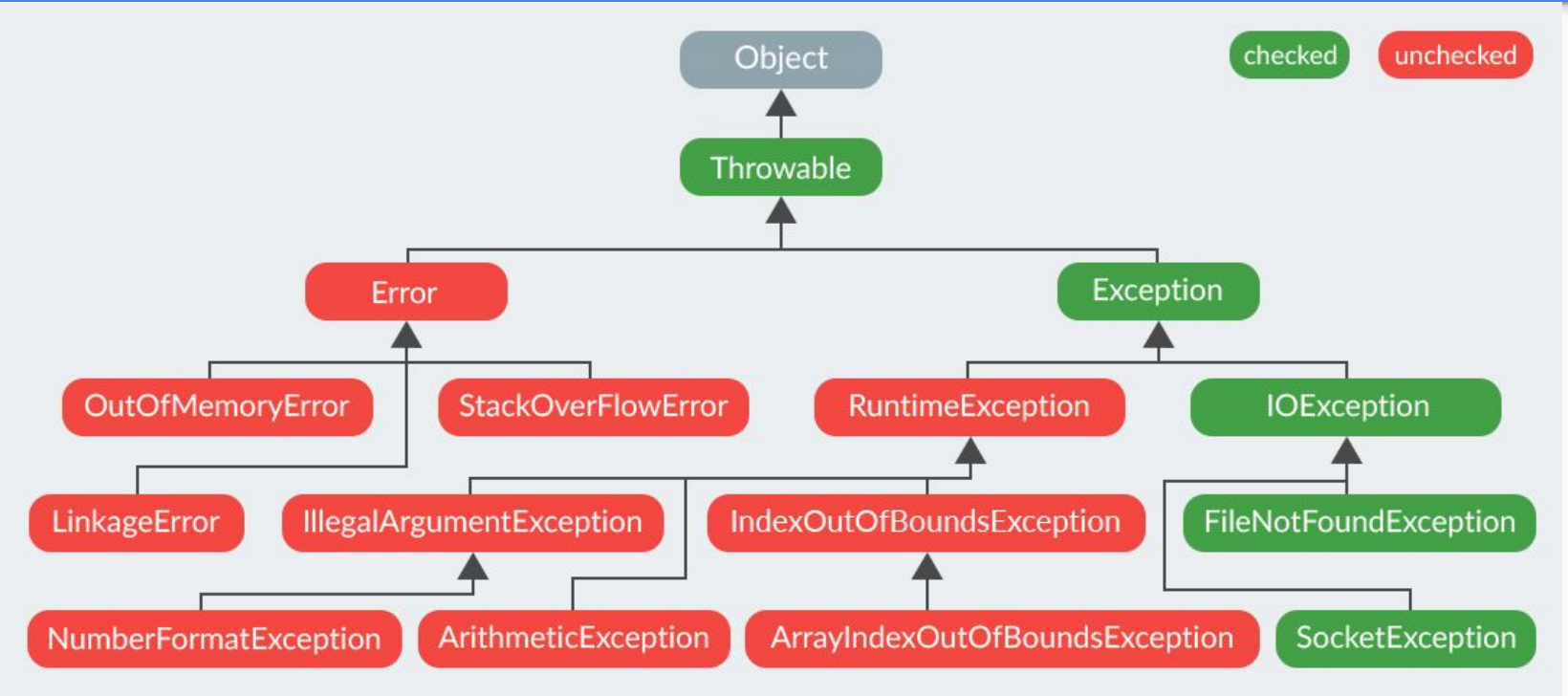
Иерархия исключений



Типы ошибок

- OutOfMemoryError
- StackOverflowError
- IllegalArgumentException
- ArrayIndexOutOfBoundsException
- FileNotFoundException
- IOException
- ClassNotFoundException
- NoSuchElementException

Проверяемые и непроверяемые



Как не нужно делать

throw new NullPointerException(“invalid”)

(это ошибка разработчика)

try-catch

```
public static void main(String[] args){
    String text = "Hello world!";

    try {
        Integer number = Integer.parseInt(text);
    } catch (NumberFormatException e) {
        e.printStackTrace();
    }
}
```

try-catch

Не транзакционный, то есть не “откатывает” изменения, сделанные в блоке try

Исключения в сигнатуре

Используются для того чтобы не обрабатывать ошибку в методе, а сообщить разработчику что в этом методе может возникнуть такая ситуация.

```
public static int parseNumber(String text) throws NumberFormatException {
    return Integer.parseInt(text);
}
```

Собственные исключения

```
public class MyOwnException extends Exception
{
    private final long exceptionTime;

    public MyOwnException(String message, long timeMillis) {
        super(message);
        this.exceptionTime = timeMillis;
    }

    public long getExceptionTime()
    {
        return exceptionTime;
    }
}
```

re-throw

```
public static int parseNumber(String text) throws MyOwnException
{
    try
    {
        return Integer.parseInt(text);
    } catch (NumberFormatException e) {
        throw new MyOwnException("Cannot parse number", System.currentTimeMillis());
    }
}
```

Closable / Autoclosable

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

finally

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

Каскад try-catch

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

try-with-resources

```
public static void modifyFile(File file) {
    try (BufferedReader bufferedReader = new BufferedReader(new FileReader(file))) {
        // do something useful
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Reflection

- Узнать класс объекта
- Получить информацию о модификаторах, полях, методах, конструкторах и иерархии классов
- Создать экземпляр неизвестного класса
- Модифицировать модификаторы
- Модифицировать поля по имени
- Вызвать метод по имени и/или сигнатуре