**Как работают исключения**

[Java Syntax](https://javarush.ru/quests/QUEST_JAVA_SYNTAX)

[Уровень 9](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=9), Лекция 5



— Хочу рассказать тебе немного о том, как работают исключения. Приведённый ниже пример будет показывать примерную логику работы:

Код с использованием исключений

class ExceptionExampleOriginal

{

public static void main(String[] args)

{

System.out.println("main begin");

try

{

System.out.println("main before call");

method1();

System.out.println("main after call");

}

catch (RuntimeException e)

{

String s = e.getMessage();

System.out.println(s);

}

System.out.println("main end");

}

public static void method1()

{

System.out.println("method1 begin");

method2();

System.out.println("method1 end");

}

public static void method2()

{

System.out.println("method2");

String s = "Message: Unknown Exception";

throw new RuntimeException(s);

}

}

Примерная расшифровка

public class ExceptionExample

{

private static Exception exception = null;

public static void main(String[] args)

{

System.out.println("main begin");

System.out.println("main before call");

method1();

if (exception == null)

{

System.out.println("main after call");

}

else if (exception instanceof RuntimeException)

{

RuntimeException e = (RuntimeException) exception;

exception = null;

String s = e.getMessage();

System.out.println(s);

}

System.out.println("main end");

}

public static void method1()

{

System.out.println("method1 begin");

method2();

if (exception != null) return;

System.out.println("method1 end");

}

public static void method2()

{

System.out.println("method2");

String s = "Message: Unknown Exception";

exception = new RuntimeException(s);

return;

}

}

— Ничего не понятно.

— Ок. Давай я тебе объясню, что происходит.

— В примере слева мы по цепочке вызываем несколько методов. В method2 мы специально создаем и выкидываем исключение (инициируем ошибку).

— В примере справа показано, что примерно при этом происходит.

— Посмотри на method2. Создание исключения превратилось вот во что: создали объект типа RuntimeException, сохранили его в специальную переменную exception и тут же вышли из метода – return.

— В методе method1, после вызова method2 стоит проверка – есть исключение или нет, если исключение есть, тогда метод method1 тут же завершается. Такая проверка неявно производится после вызова каждого(!) метода в Java.

— Ого!

— Вот тебе и ого.

— В колонке справа в методе main я написал, что примерно происходит при перехвате исключения с помощью конструкции try-catch. Если исключения не было, то все продолжает работать, как и запланировано. Если исключение было, и оно было такого типа, как указано в catch, тогда мы его обрабатываем.

— А что значит throw и instanceof?

– Посмотри на последнюю строку throw new RuntimeException(s). Таким способом мы создаем и кидаем исключение. Пока мы так делать не будем. Это только для примера.

— А с помощью команды «а instanceof B» мы проверяем, имеет ли объект a тип B. Т.е. имеет ли объект, который хранится в переменной exception, тип RuntimeException. Это логическое условие.

— Ну, вроде понятно. Немного.