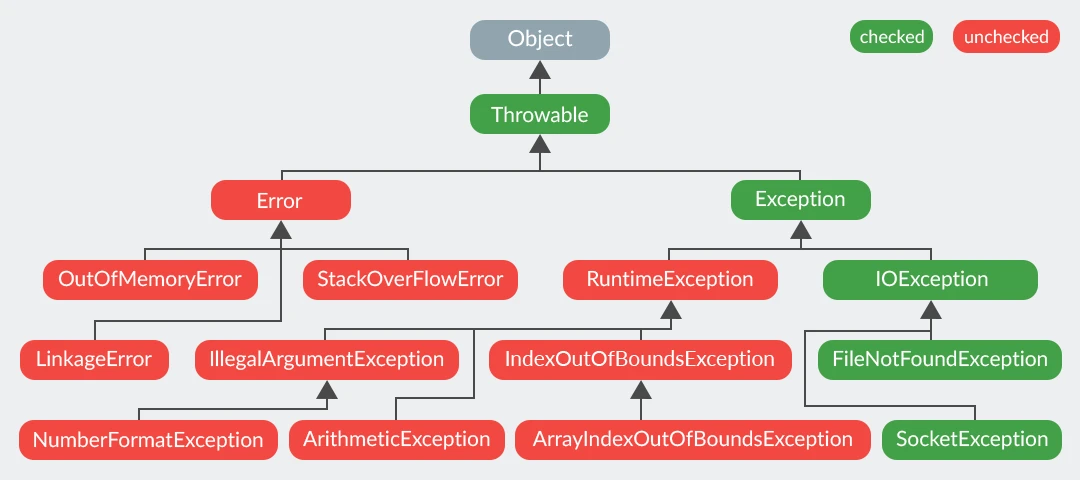
**Типы исключений**

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

[Уровень 9](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=9), Лекция 4

[](https://cdn.javarush.ru/images/article/09750c19-a2a4-4183-aa61-95dc3546404a/original.jpeg)

— Решила поднять сегодня ещё одну тему. В Java все исключения делятся на два типа – контролируемые/проверяемые (checked) и неконтролируемые/непроверяемые (unchecked): те, которые перехватывать обязательно, и те, которые перехватывать не обязательно. По умолчанию – все исключения обязательно нужно перехватывать.

— А можно в коде специально выбрасывать исключения?

— В своем коде ты сам можешь выкидывать исключения. Ты даже можешь написать свои собственные исключения. Но это мы разберем позже. Сейчас же давай научимся работать с исключениями, которые выбрасывает Java-машина.

— Ок.

— Если в методе выбрасываются (возникают) исключения ClassNotFoundException и FileNotFoundException, программист обязан указать их в сигнатуре метода (в заголовке метода). Это checked исключения. Вот как это обычно выглядит:

|  |
| --- |
| **Примеры** |
| public static void method1() throws ClassNotFoundException, FileNotFoundException |
| public static void main() throws IOException |
| public static void main() //не выбрасывает никаких исключений |

— Т.е. мы просто пишем throws и перечисляем исключения через запятую. Так?

— Да. Но тут интересно другое. Чтобы программа скомпилировалась, метод, который вызывает method1 в примере ниже, должен сделать две вещи: или перехватить эти исключения или пробросить их дальше (тому, кто его вызвал), указав их в своём заголовке.

— Еще раз. Если ты в методе main хочешь вызвать метод какого-то объекта, в заголовке которого прописано throws FileNotFoundException, … то тебе надо сделать одно из двух:

1) Перехватывать исключения FileNotFoundException, …

Тебе придется обернуть код вызова опасного метода в блок **try-catch**

2) Не перехватывать исключения FileNotFoundException, …

Тебе придется добавить эти исключения в список **throws** своего метода **main**.

— А можно пример?

— Вот, смотри:

Примеры проверяемых (checked) исключений

public static void main(String[] args)

{

method1();

}

public static void method1() throws FileNotFoundException, ClassNotFoundException

{

//тут кинется исключение FileNotFoundException, такого файла нет

FileInputStream fis = new FileInputStream("C2:\badFileName.txt");

}

— Этот пример не скомпилируется, т.к. метод **main** вызывает метод method1(), который выкидывает исключения, обязательные к перехвату.

— Чтобы пример скомпилировался, надо добавить обработку исключений в метод main. Сделать это можно двумя способами:

Способ 1: просто пробрасываем исключение выше (вызывающему):

public static void main(String[] args) throws FileNotFoundException, ClassNotFoundException

{

method1();

}

public static void method1() throws FileNotFoundException, ClassNotFoundException

{

//тут кинется исключение FileNotFoundException, такого файла нет

FileInputStream fis = new FileInputStream("C2:\badFileName.txt");

}

— А тут перехватываем его с помощью **try-catch**:

Способ 2: перехватываем исключение:

public static void main(String[] args)

{

try

{

method1();

}

catch(Exception e)

{

}

}

public static void method1() throws FileNotFoundException, ClassNotFoundException

{

//тут кинется исключение FileNotFoundException, такого файла нет

FileInputStream fis = new FileInputStream("C2:\badFileName.txt");

}

— Что-то понемногу проясняется.

— Посмотри на пример ниже, чтобы разобраться:

Не обрабатываем исключения – нужно пробросить их дальше, тому, кто знает как

public static void method2() throws FileNotFoundException, ClassNotFoundException

{

method1();

}

Обрабатываем одно исключение, второе – пробрасываем:

public static void method3() throws ClassNotFoundException

{

try

{

method1();

}

catch (FileNotFoundException **e**)

{

System.out.println("FileNotFoundException has been caught.");

}

}

Перехватываем оба – ничего не пробрасываем:

public static void method4()

{

try

{

method1();

}

catch (FileNotFoundException **e**)

{

System.out.println("FileNotFoundException has been caught.");

}

catch (ClassNotFoundException **e**)

{

System.out.println("ClassNotFoundException has been caught.");

}

}

3

Задача

Java Syntax,  9 уровень,  4 лекция

Набираем код Ӏ Java Syntax: 9 уровень, 4 лекция

Java Syntax: 9 уровень, 4 лекция. Внимание! Объявляется набор кода на JavaRush. Для этого включите режим повышенной внимательности, расслабьте пальцы, читайте код и… набирайте его в соответствующем окошке. Набор кода — вовсе не бесполезное занятие, как может показаться на первый взгляд: благодаря ему новичок привыкает к синтаксису и запоминает его (современные IDE редко дают ему это сделать).

— Но есть вид исключений – это *RuntimeException* и классы, унаследованные от него. Их перехватывать не обязательно. Это unchecked исключения. Считается, что это трудно прогнозируемые исключения и предсказать их появление практически невозможно. С ними можно делать все то же самое, но указывать в **throws** их не нужно.