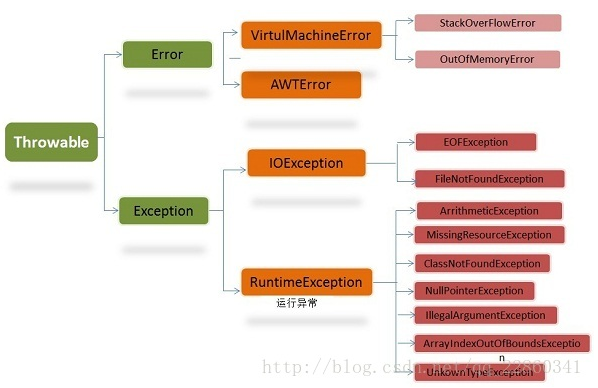
Unchecked exception 与 Checked exeception区别

# Java异常类结构图



# Throwable类、Error类、Exception类介绍：

## Throwable类

public class **Throwable** extends **Object** implements **Serializable**

**一直错误地以为Throwable是一个接口，其实是一个类，其下有两个子类：Error和Exception。**

java.lang.Object

java.lang.**Throwable**

Direct Known Subclasses: **Error, Exception**

## Error类

常见的Error类：

**Direct Known Subclasses:**

[AnnotationFormatError](https://docs.oracle.com/javase/8/docs/api/java/lang/annotation/AnnotationFormatError.html), [AssertionError](https://docs.oracle.com/javase/8/docs/api/java/lang/AssertionError.html), [AWTError](https://docs.oracle.com/javase/8/docs/api/java/awt/AWTError.html), [CoderMalfunctionError](https://docs.oracle.com/javase/8/docs/api/java/nio/charset/CoderMalfunctionError.html), [FactoryConfigurationError](https://docs.oracle.com/javase/8/docs/api/javax/xml/parsers/FactoryConfigurationError.html), [FactoryConfigurationError](https://docs.oracle.com/javase/8/docs/api/javax/xml/stream/FactoryConfigurationError.html),[IOError](https://docs.oracle.com/javase/8/docs/api/java/io/IOError.html), [LinkageError](https://docs.oracle.com/javase/8/docs/api/java/lang/LinkageError.html), [SchemaFactoryConfigurationError](https://docs.oracle.com/javase/8/docs/api/javax/xml/validation/SchemaFactoryConfigurationError.html), [ServiceConfigurationError](https://docs.oracle.com/javase/8/docs/api/java/util/ServiceConfigurationError.html), [ThreadDeath](https://docs.oracle.com/javase/8/docs/api/java/lang/ThreadDeath.html),[TransformerFactoryConfigurationError](https://docs.oracle.com/javase/8/docs/api/javax/xml/transform/TransformerFactoryConfigurationError.html), [**VirtualMachineError**](https://docs.oracle.com/javase/8/docs/api/java/lang/VirtualMachineError.html)

**重点是**[**VirtualMachineError**](https://docs.oracle.com/javase/8/docs/api/java/lang/VirtualMachineError.html)**虚拟机错误，其下有Error：**

**Direct Known Subclasses:**

[InternalError](https://docs.oracle.com/javase/8/docs/api/java/lang/InternalError.html), [OutOfMemoryError](https://docs.oracle.com/javase/8/docs/api/java/lang/OutOfMemoryError.html), [StackOverflowError](https://docs.oracle.com/javase/8/docs/api/java/lang/StackOverflowError.html), [UnknownError](https://docs.oracle.com/javase/8/docs/api/java/lang/UnknownError.html)

## Exception

public class **Exception** extends **Throwable**

**其中有重要一类就是RuntimeException**

public class **RuntimeException** extends [Exception](https://docs.oracle.com/javase/8/docs/api/java/lang/Exception.html)

**RuntimeException**：Unchecked Exception

最常见的就是空指针异常**NullPointerException**、**ClassCastException类转换异常、IndexOutOfBoundsException指针越界异常。**

Direct Known Subclasses:

AnnotationTypeMismatchException, ArithmeticException, ArrayStoreException, **BufferOverflowException**, **BufferUnderflowException**, CannotRedoException, CannotUndoException, **ClassCastException**, CMMException, CompletionException, ConcurrentModificationException, DataBindingException, DateTimeException, DOMException, EmptyStackException, EnumConstantNotPresentException, EventException, FileSystemAlreadyExistsException, FileSystemNotFoundException, IllegalArgumentException, IllegalMonitorStateException, **IllegalPathStateException**, **IllegalStateException**, IllformedLocaleException, ImagingOpException, IncompleteAnnotationException, **IndexOutOfBoundsException**, JMRuntimeException, LSException, MalformedParameterizedTypeException, MalformedParametersException, MirroredTypesException, **MissingResourceException**, NegativeArraySizeException, **NoSuchElementException**, NoSuchMechanismException, **NullPointerException**, ProfileDataException, ProviderException, ProviderNotFoundException, RasterFormatException, RejectedExecutionException, SecurityException, SystemException, TypeConstraintException, TypeNotPresentException, UncheckedIOException, UndeclaredThrowableException, UnknownEntityException, UnmodifiableSetException, UnsupportedOperationException, WebServiceException, WrongMethodTypeException

# 介绍

**Throwable接口**： 有两个重要的子类：**Exception（异常）和 Error（错误），**二者都是 Java 异常处理的重要子类，各自都包含大量子类。

**Error（错误）**:是程序无法处理的错误，表示运行应用程序中较严重问题。大多数错误与代码编写者执行的操作无关，而表示代码运行时 **JVM（Java 虚拟机）出现**的问题。例如，Java虚拟机运行错误（Virtual MachineError），当 JVM 不再有继续执行操作所需的内存资源时，将出现 **OutOfMemoryError**。这些异常发生时，Java虚拟机（JVM）一般会选择线程终止。

**Exception（异常）**:是程序本身可以处理的异常。

Exception可以分为**检查型异常（Checked Exception）**与**非检查型异常(运行时异常)**（Unchecked Exception）。

## Unchecked Exception：运行时异常RuntimeException及其子类异常

**unchecked exceptions（运行时异常）都是RuntimeException类及其子类异常，就是我们在开发中测试功能时程序终止，控制台出现的异常，**比如：

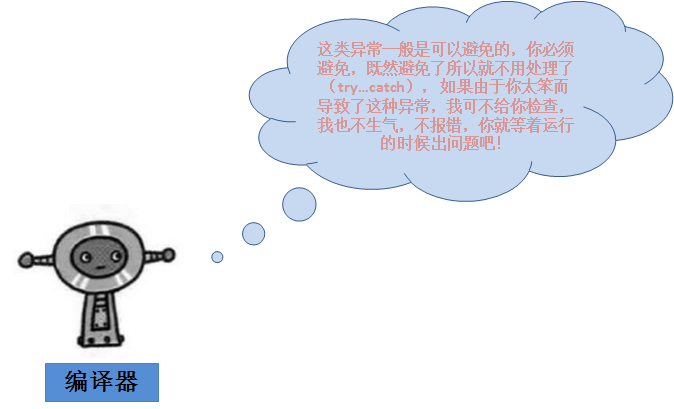
**NullPointerException(空指针异常)、 IndexOutOfBoundsException(下标越界异常) 、**

**ClassCastException(类转换异常) 、ArrayStoreException(数据存储异常，操作数组时类型不一致) 、NIO操作的BufferOverflowException、BufferUnderflowException异常(注意IO异常一般是检查型异常)。**

**运行时异常--非检查型异常（Unchecked Exception）**

**个人理解：**所谓非检查（**Unchecked**）是指**编译器**不会检查这类异常，不检查的则开发者在代码的编辑编译阶段就不是必须处理，这类异常一般可以避免，因此无需处理（try ...catch）。如果不处理这类异常，集成开发环境中的编译器也不会给出错误提示。

**例如：**你的程序逻辑本身有问题，比如数组越界、访问null对象，这种错误你自己是可以避免的。**编译器不会强制你检查这种异常**。



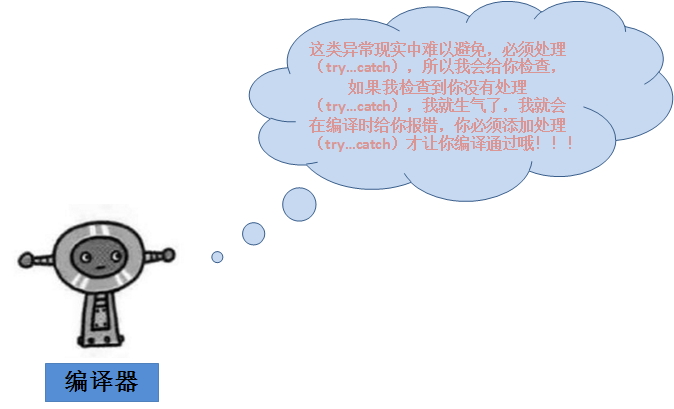
## Checked Exception检查型异常

checked exceptions，非运行时异常 （编译异常）：是**RuntimeException以外的异常**，类型上都属于**Exception类及其子类**。**从程序语法角度讲是必须进行处理的异常，如果不处理，程序就不能编译通过。**如IOException、SQLException等以及用户自定义的Exception异常，一般情况下不自定义检查异常。

通俗的话说就是在写代码时出现红线，需要try catch或者throws时出现的异常。

**个人理解：**所谓检查（**Checked**）是指**编译器**要检查这类异常，检查的目的一方面是因为该类异常的发生难以避免，另一方面就是让开发者去解决掉这类异常，**所以称为必须处理（try ...catch）的异常**。如果不处理这类异常，集成开发环境中的编译器一般会给出错误提示。

**例如：**一个读取文件的方法代码逻辑没有错误，但程序运行时可能会因为文件找不到而抛出**FileNotFoundException**，如果不处理这些异常，程序将来肯定会出错。所以编译器会提示你要去捕获并处理这种可能发生的异常，不处理就不能通过编译。



# 关于异常的题

## 运行时异常与一般异常有何异同？

**异常**表示程序运行过程中**可能出现的非正常状态**，运行时异常表示**虚拟机的通常操作中可能遇到的异常，是一种常见运行错误**。java编译器要求方法必须声明抛出可能发生的非运行时异常(检查型异常Checked Exception)，但是并不要求必须声明抛出未被捕获的运行时异常。

## Error和Exception有什么区别?

**Error 表示恢复不是不可能但很困难的情况下的一种严重问题**。比如说内存溢出。不可能指望程序能处理这样的情况。 exception 表示一种设计或实现问题。也就是说，它表示如果程序运行正常，从不会发生的情况。

## Java中的异常处理机制的简单原理和应用。

异常是指java程序运行时（非编译）所发生的非正常情况或错误，与现实生活中的事件很相似，现实生活中的事件可以包含事件发生的时间、地点、人物、情节等信息，可以用一个对象来表示，**Java使用面向对象的方式来处理异常，它把程序中发生的每个异常也都分别封装到一个对象来表示的，该对象中包含有异常的信息。**

Java对异常进行了分类，不同类型的异常分别用不同的Java类表示，所有异常的根类为**java.lang.Throwable**，**Throwable下面又派生了两个子类：Error和Exception**，Error 表示应用程序本身无法克服和恢复的一种严重问题，程序只有死的份了，例如，说内存溢出和线程死锁等系统问题。Exception表示程序还能够克服和恢复的问题，其中又分为系统异常和普通异常，系统异常是软件本身缺陷所导致的问题，也就是软件开发人员考虑不周所导致的问题，软件使用者无法克服和恢复这种问题，但在这种问题下还可以让软件系统继续运行或者让软件死掉，例如，数组脚本越界（**ArrayIndexOutOfBoundsException**），空指针异常（**NullPointerException**）、类转换异常（**ClassCastException**）；普通异常是运行环境的变化或异常所导致的问题，是用户能够克服的问题，例如，网络断线，硬盘空间不够，发生这样的异常后，程序不应该死掉。

java为系统异常和普通异常提供了不同的解决方案，编译器强制普通异常必须**try..catch处理或用throws声明继续抛给上层调用方法处理**，所以**普通异常也称为checked异常**，而系统异常可以处理也可以不处理，所以，**编译器不强制用try..catch处理或用throws声明，所以系统异常也称为unchecked异常**。

提示答题者：就按照三个级别去思考：**虚拟机必须宕机的错误**，**程序可以死掉也可以不死掉的错误**，**程序不应该死掉的错误**。

## 请写出你最常见到的5个runtime exception。

**NullPointerException、ClassNotFoundException、**ClassCastException、**FileNotFoundException、ArrayIdexOutOfBoundsException。**

**ArithmeticException**——由于除数为0引起的异常；

**ArrayStoreException**——由于数组存储空间不够引起的异常；

ClassCastException—一当把一个对象归为某个类，但实际上此对象并不是由这个类 创建的，也不是其子类创建的，则会引起异常；

**IllegalMonitorStateException**——监控器状态出错引起的异常；

**NegativeArraySizeException**—一数组长度是负数，则产生异常；

**NullPointerException**—一程序试图访问一个空的数组中的元素或访问空的对象中的 方法或变量时产生异常；

**OutofMemoryException**——用new语句创建对象时，如系统无法为其分配内存空 间则产生异常；

**SecurityException**——由于访问了不应访问的指针，使安全性出问题而引起异常；

**IndexOutOfBoundsExcention**——由于数组下标越界或字符串访问越界引起异常；

**IOException**——由于文件未找到、未打开或者I/O操作不能进行而引起异常；

**ClassNotFoundException**——未找到指定名字的类或接口引起异常；

**CloneNotSupportedException**——一程序中的一个对象引用Object类的clone方法，但 此对象并没有连接Cloneable接口，从而引起异常；

**InterruptedException**—一当一个线程处于等待状态时，另一个线程中断此线程，从 而引起异常，有关线程的内容，将在下一章讲述；

**NoSuchMethodException**一所调用的方法未找到，引起异常；

Illega1AccessExcePtion—一试图访问一个非public方法；

StringIndexOutOfBoundsException——访问字符串序号越界，引起异常；

**ArrayIndexOutOfBoundsException**—一访问数组元素下标越界，引起异常；

NumberFormatException——字符的UTF代码数据格式有错引起异常；

IllegalThreadException—一线程调用某个方法而所处状态不适当，引起异常；

**FileNotFoundException**——未找到指定文件引起异常；

**EOFException**——未完成输入操作即遇文件结束引起异常。

# 5种最常见的异常

在Java学习过程中，我们会碰到各种各样的异常，现将面试过程中遇到的常见异常做一罗列并举例：

## NullPointerException: 空指针异常。菜鸟编程经验发现这个异常是经常会发生的，属于运行时异常。

举例：

public static void main(String[] args) {

String str = null;

//此处报空指针异常

System.out.println(str.length());

}

控制台输出的异常信息为：

Exception in thread "main" java.lang.NullPointerException

at cn.com.gjw.MyClass.main(MyClass.java:7)

## ClassCastException: 类型强制转换异常。Java的一些语法知识不太熟悉的时候会发生的异常。

举例：

public static void main(String[] args) {

// 类型强制转换异常

Object x = new String("String");

System.out.println((Integer) x);

}

控制台输出的异常信息为：

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

at cn.com.gjw.MyClass.main(MyClass.java:7)

## ArrayIndexOutOfBoundsException: 数组下标越界异常。特别常见的一个异常，粗心的程序员都会犯的错误。

举例：

public static void main(String[] args) {

int arr[] = {1,2};

// 此处报数组下标越界异常

System.out.println(arr[2]);

}

控制台输出的异常信息为：

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 2

at cn.com.gjw.MyClass.main(MyClass.java:7)

## ArithmeticException:算术运算异常。算术跟我水平差不多的你有没有犯过类似的错误呢？？

举例：

public static void main(String[] args) {

// 整数0做了分母，报算术运算异常

System.out.println(1 / 0);

}

控制台输出的异常信息为：

Exception in thread "main" java.lang.ArithmeticException: / by zero

at cn.com.gjw.MyClass.main(MyClass.java:6)

## NumberFormatException: 数字格式异常。什么是数字格式异常呢？一个例子说明一切。

举例：

public static void main(String[] args) {

// 将字符串“it”转换为Integer类型的，当然会报数字格式异常啦

System.out.println(Integer.parseInt("it"));

}

控制台输出的异常信息为：

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

at java.lang.NumberFormatException.forInputString(NumberFormatException.java:65)

at java.lang.Integer.parseInt(Integer.java:492)

at java.lang.Integer.parseInt(Integer.java:527)

at cn.com.gjw.MyClass.main(MyClass.java:6)