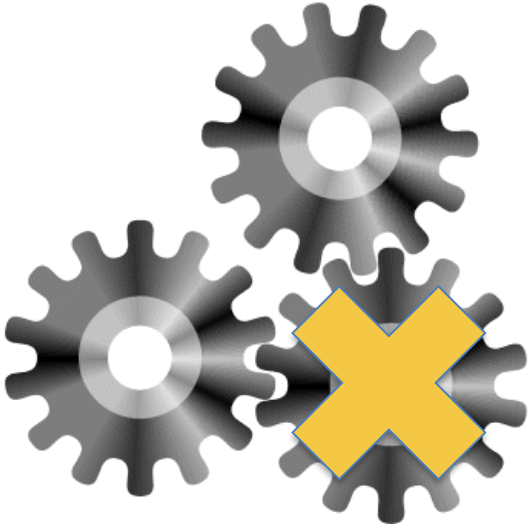
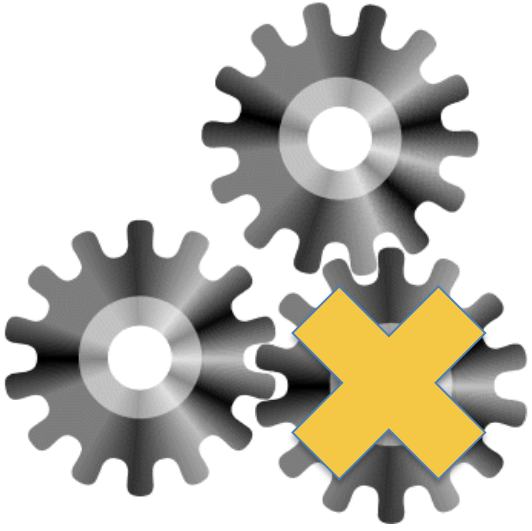
Exceptions: Introduction

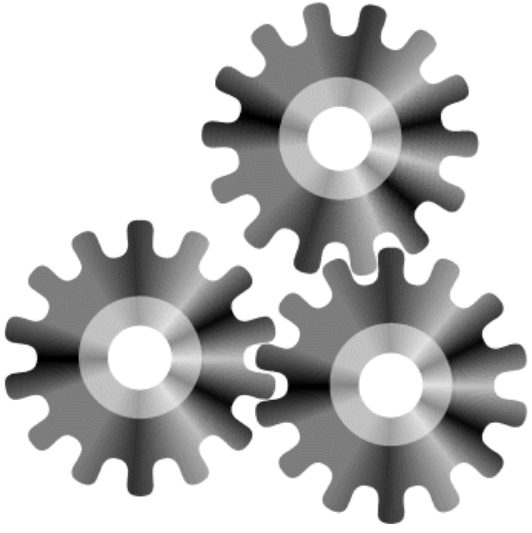
1. Exceptions: Introduction
2. Exception: Types

Exceptions: Introduction

1. Exceptions: Introduction

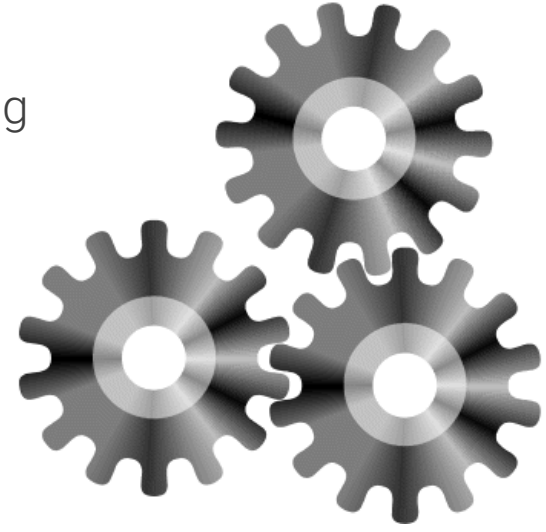






Something is going wrong

`throw`



Do something to keep the application working.

`catch`

Error vs Exception

Error	Exception
Irrecoverable situation	Recoverable situation
Related to environment	Related to application
There is no programable fix	May be anticipated and should be handled

Error vs Exception

Error	Exception
Irrecoverable situation	Recoverable situation
Related to environment	Related to application
There is no programable fix	May be anticipated and should be handled

`OutOfMemoryError`

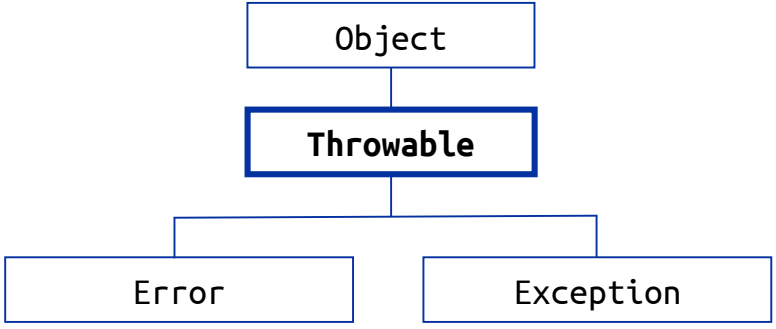
Error vs Exception

Error	Exception
Irrecoverable situation	Recoverable situation
Related to environment	Related to application
There is no programable fix	May be anticipated and should be handled
`OutOfMemoryError`	`ArrayIndexOutOfBoundException`

Error vs Exception

Error	Exception
Irrecoverable situation	Recoverable situation
Related to environment	Related to application
There is no programable fix	May be anticipated and should be handled
`OutOfMemoryError`	`ArrayIndexOutOfBoundException`
`class Error`	`class Exception`

Class Throwable



Error and Exception classes inherit from Throwable superclass

Class Throwable

Constructors

Modifier	Constructor and Description
	`Throwable()` Constructs a new throwable with `null` as its detail message.
	`Throwable(String message)` Constructs a new throwable with the specified detail message.
	`Throwable(String message, Throwable cause)` Constructs a new throwable with the specified detail message and cause.
`protected`	`Throwable(String message, Throwable cause, boolean enableSuppression, boolean writableStackTrace)` Constructs a new throwable with the specified detail message, cause, **suppression** enabled or disabled, and writable stack trace enabled or disabled.
	`Throwable(Throwable cause)` Constructs a new throwable with the specified cause and a detail message of `(cause==null ? null : cause.toString())` (which typically contains the class and detail message of `cause`).

https://docs.oracle.com/javase/7/docs/api/java/lang/Throwable.html

Class Throwable

Methods

Modifier and Type	Method and Description
`void`	**`addSuppressed(Throwable exception)`** Appends the specified exception to the exceptions that were suppressed in order to deliver this exception.
`Throwable`	**`fillInStackTrace()`** Fills in the execution stack trace.
`Throwable`	**`getCause()`** Returns the cause of this throwable or `null` if the cause is nonexistent or unknown.
`String`	**`getLocalizedMessage()`** Creates a localized description of this throwable.
`String`	**`getMessage()`** Returns the detail message string of this throwable.
`StackTraceElement[]`	**`getStackTrace()`** Provides programmatic access to the stack trace information printed by **`printStackTrace()`**.
`Throwable[]`	**`getSuppressed()`** Returns an array containing all of the exceptions that were suppressed, typically by the `try`-with-resources statement, in order to deliver this exception.
`Throwable`	**`initCause(Throwable cause)`** Initializes the *cause* of this throwable to the specified value.
`void`	**`printStackTrace()`** Prints this throwable and its backtrace to the standard error stream.
`void`	**`printStackTrace(PrintStream s)`** Prints this throwable and its backtrace to the specified print stream.
`void`	**`printStackTrace(PrintWriter s)`** Prints this throwable and its backtrace to the specified print writer.
`void`	**`setStackTrace(StackTraceElement[] stackTrace)`** Sets the stack trace elements that will be returned by **`getStackTrace()`** and printed by **`printStackTrace()`** and related methods.
`String`	**`toString()`** Returns a short description of this throwable.

Class Throwable

Methods

Modifier and Type	Method and Description
`void`	**`addSuppressed(Throwable exception)`** Appends the specified exception to the exceptions that were suppressed in order to deliver this exception.
`Throwable`	**`fillInStackTrace()`** Fills in the execution stack trace.
`Throwable`	**`getCause()`** Returns the cause of this throwable or `null` if the cause is nonexistent or unknown.
`String`	**`getLocalizedMessage()`** Creates a localized description of this throwable.
`String`	**`getMessage()`** Returns the detail message string of this throwable.
`StackTraceElement[]`	**`getStackTrace()`** Provides programmatic access to the stack trace information printed by **`printStackTrace()`**.
`Throwable[]`	**`getSuppressed()`** Returns an array containing all of the exceptions that were suppressed, typically by the `try`-with-resources statement, in order to deliver this exception.
`Throwable`	**`initCause(Throwable cause)`** Initializes the *cause* of this throwable to the specified value.
`void`	**`printStackTrace()`** Prints this throwable and its backtrace to the standard error stream.
`void`	**`printStackTrace(PrintStream s)`** Prints this throwable and its backtrace to the specified print stream.
`void`	**`printStackTrace(PrintWriter s)`** Prints this throwable and its backtrace to the specified print writer.
`void`	**`setStackTrace(StackTraceElement[] stackTrace)`** Sets the stack trace elements that will be returned by **`getStackTrace()`** and printed by **`printStackTrace()`** and related methods.
`String`	**`toString()`** Returns a short description of this throwable.

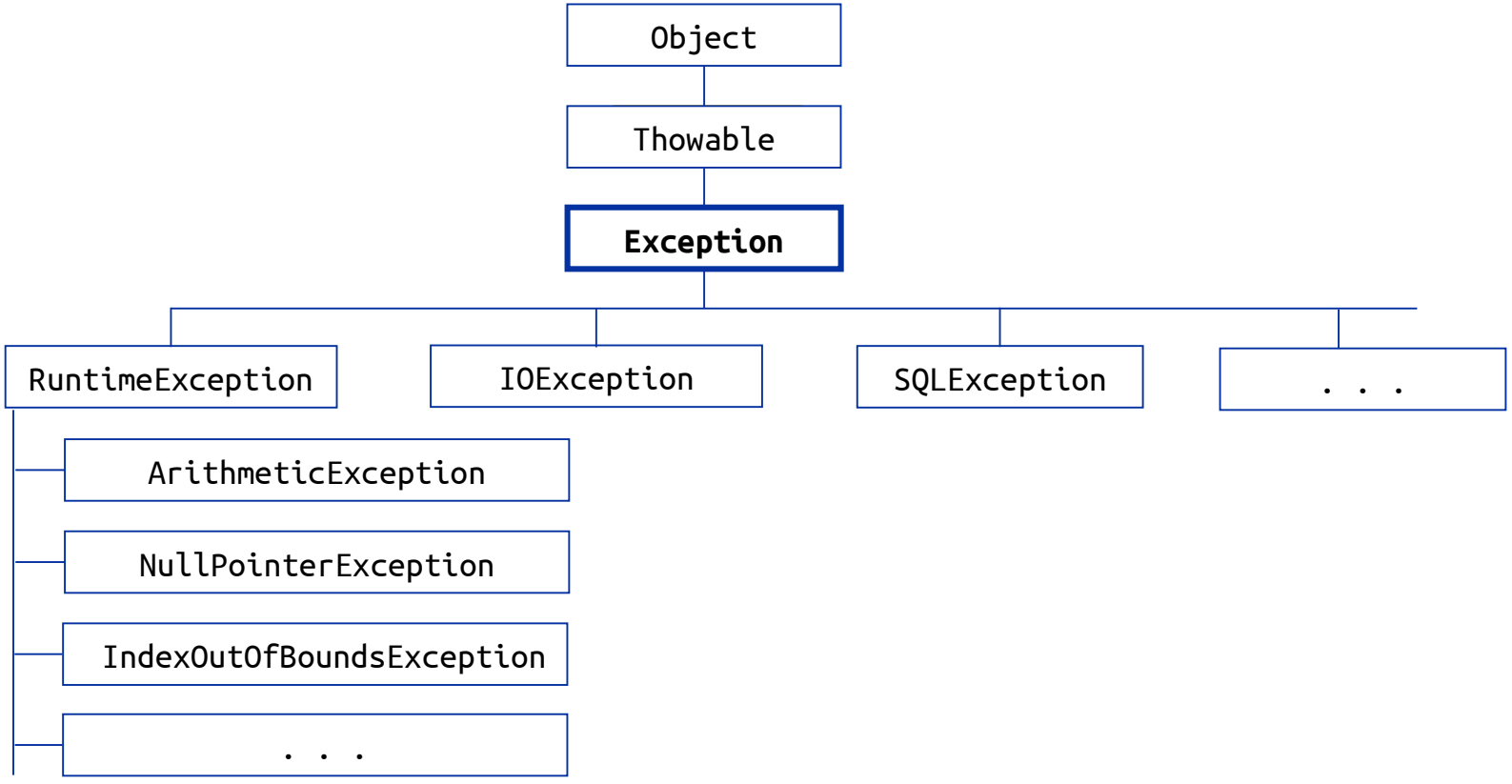
Class Throwable

Methods

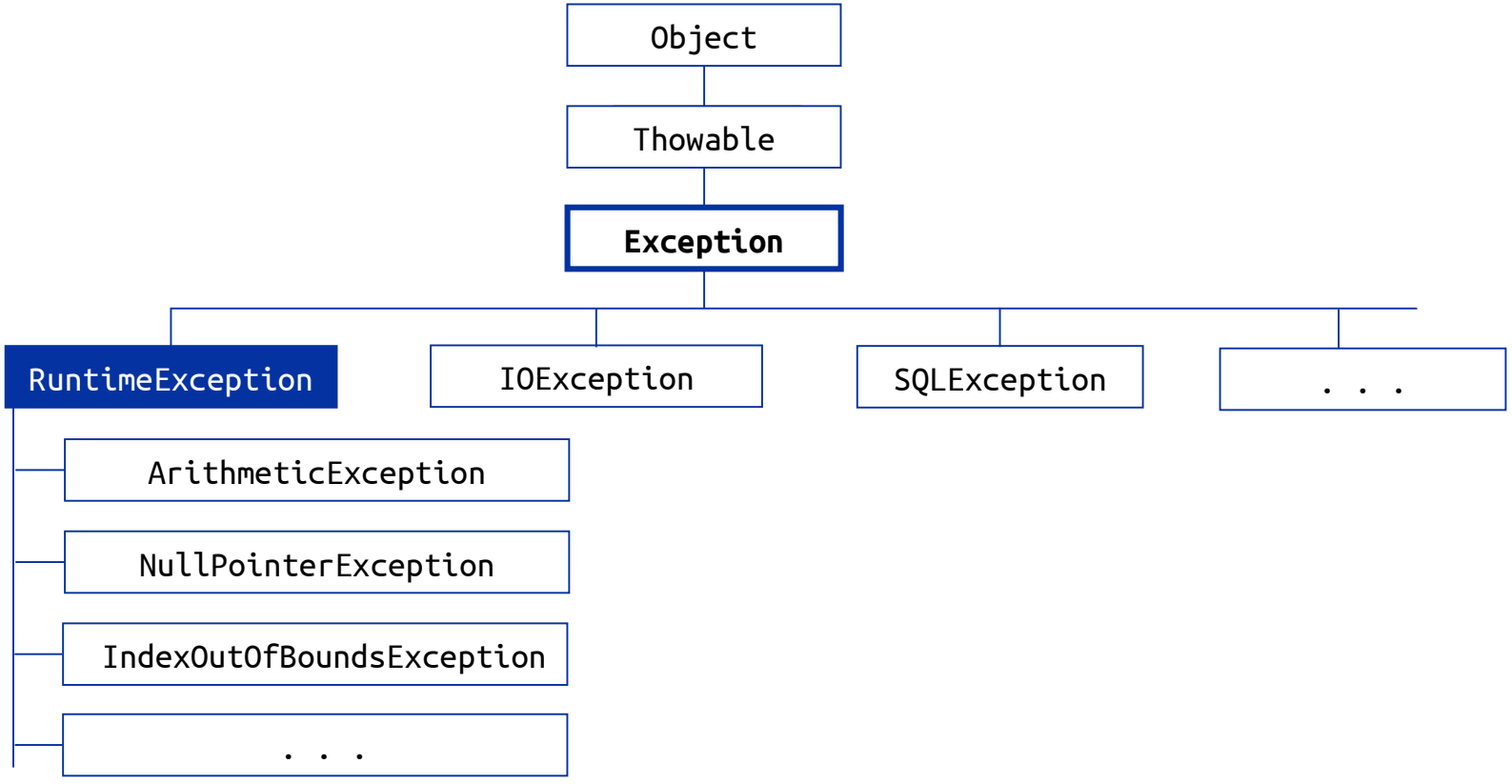
Modifier and Type	Method and Description
void	**addSuppressed**(**Throwable** exception) Appends the specified exception to the exceptions that were suppressed in order to deliver this exception.
Throwable	**fillInStackTrace**() Fills in the execution stack trace.
Throwable	**getCause**() Returns the cause of this throwable or `null` if the cause is nonexistent or unknown.
String	**getLocalizedMessage**() Creates a localized description of this throwable.
String	**getMessage**() Returns the detail message string of this throwable.
StackTraceElement[]	**getStackTrace**() Provides programmatic access to the stack trace information printed by **printStackTrace()**.
Throwable[]	**getSuppressed**() Returns an array containing all of the exceptions that were suppressed, typically by the `try`-with-resources statement, in order to deliver this exception.
Throwable	**initCause**(**Throwable** cause) Initializes the *cause* of this throwable to the specified value.
void	**printStackTrace**() Prints this throwable and its backtrace to the standard error stream.
void	**printStackTrace**(**PrintStream** s) Prints this throwable and its backtrace to the specified print stream.
void	**printStackTrace**(**PrintWriter** s) Prints this throwable and its backtrace to the specified print writer.
void	**setStackTrace**(**StackTraceElement**[] stackTrace) Sets the stack trace elements that will be returned by **getStackTrace()** and printed by **printStackTrace()** and related methods.
String	**toString**() Returns a short description of this throwable.

Exceptions: Types

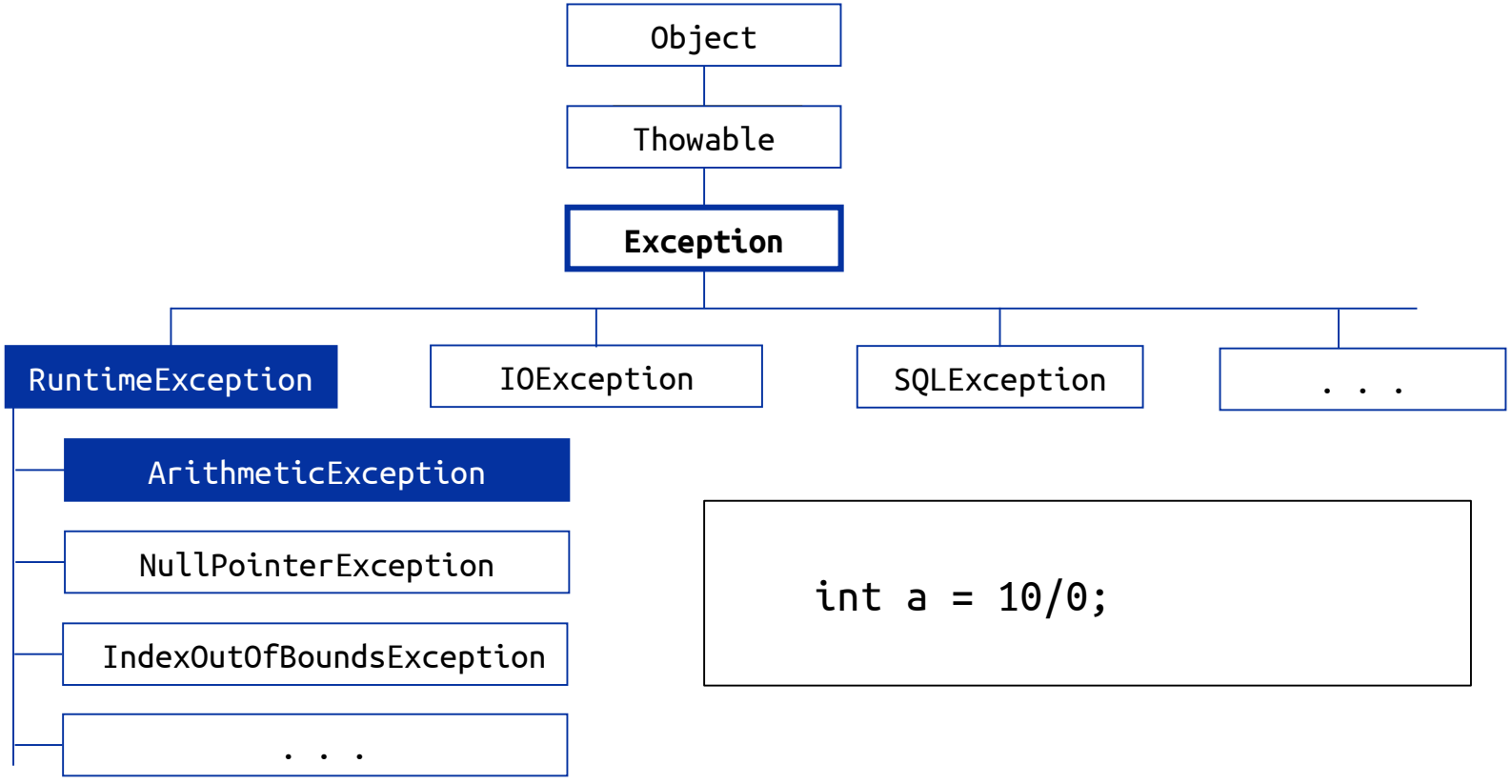
2. Exceptions: Types



2. Exceptions: Types

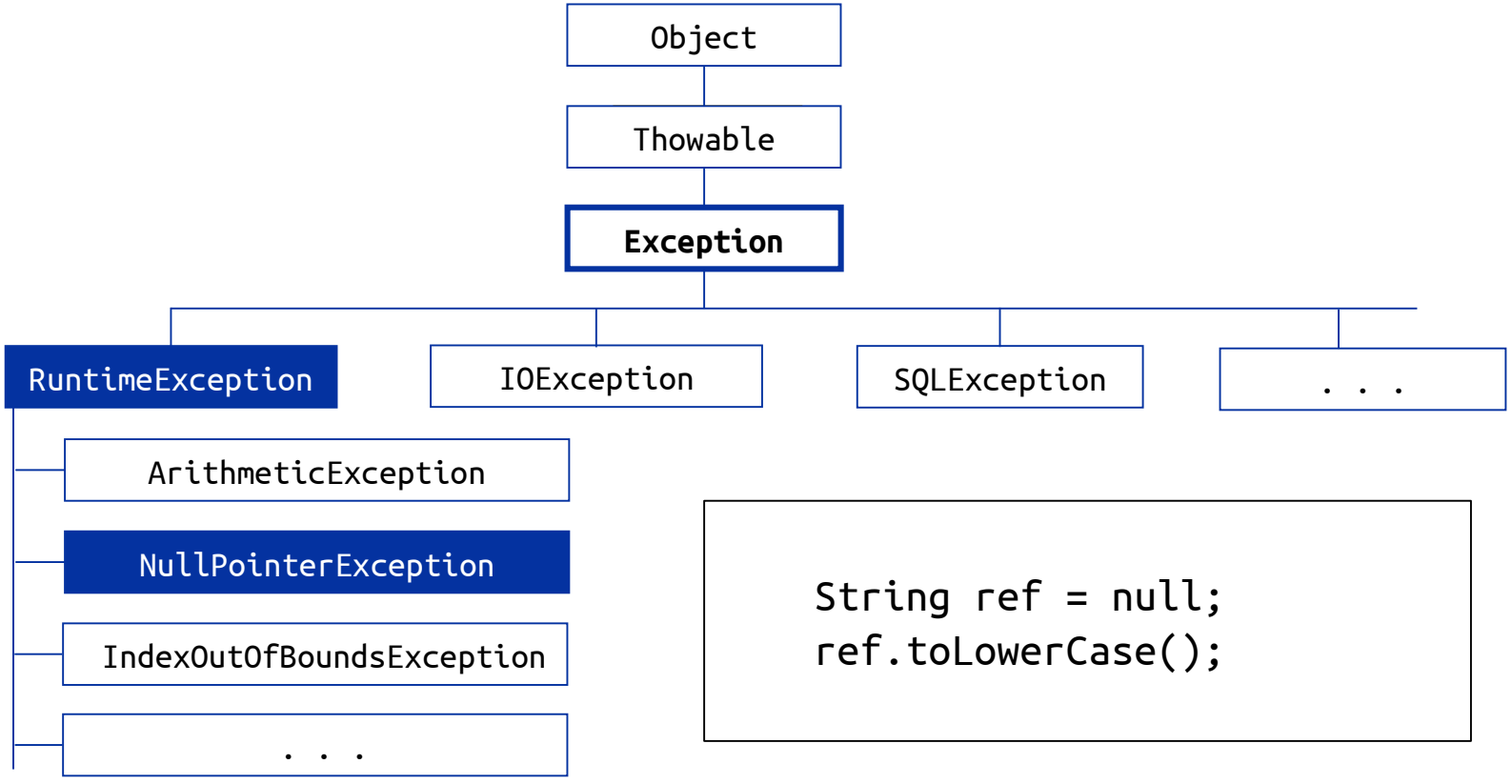


2. Exceptions: Types



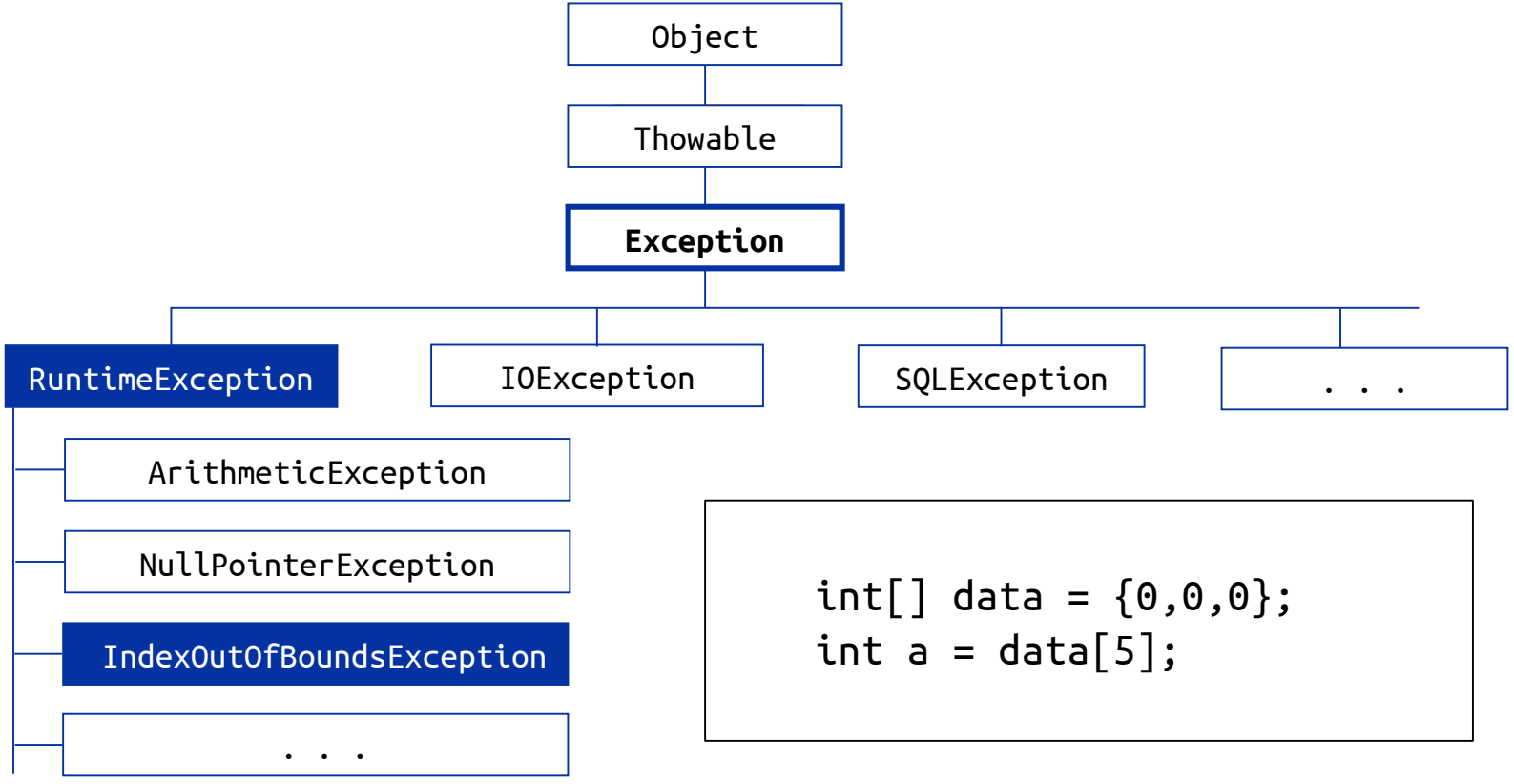

```
int a = 10/0;
```

2. Exceptions: Types



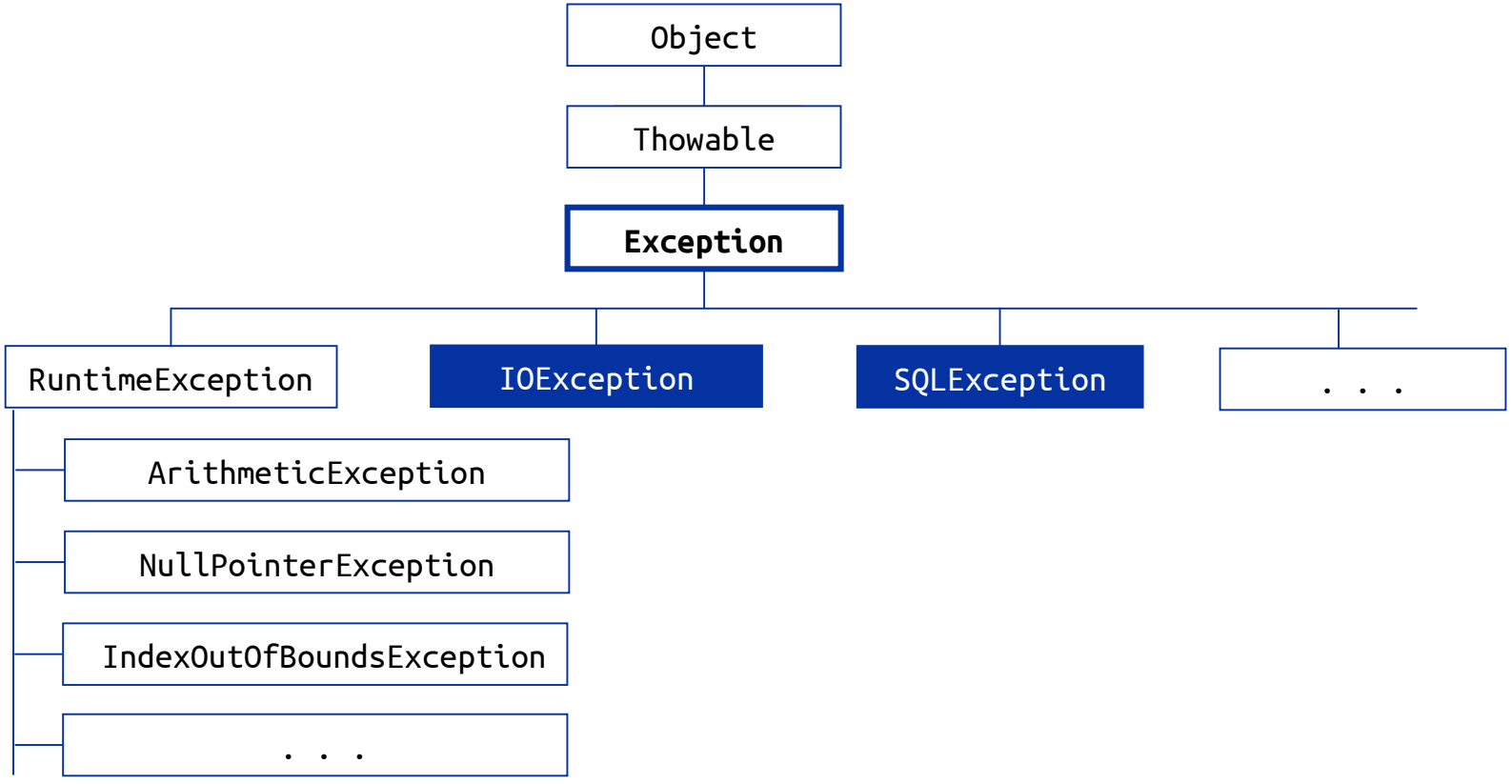

```
String ref = null;
ref.toLowerCase();
```

2. Exceptions: Types




```
int[] data = {0,0,0};
int a = data[5];
```

2. Exceptions: Types



Implicit vs Explicit

Implicit	Explicit
RunTimeExceptions	Not RunTimeExceptions
May/Should be controlled	Must be controlled
Can't be worked around in run time	May be worked around in run time

Implicit vs Explicit

Implicit	Explicit
RunTimeExceptions	Not RunTimeExceptions
May/Should be controlled	Must be controlled
Can't be worked around in run time	May be worked around in run time

Implicit Exception

Example

```
Public class Main {
    public static void main (string[]){
        String filename=null;
        if (filename.contains("aaaa")){
            //TODO code application
        }else{
            //TODO code application
        }
    }
```

Implicit Exception

Example

```
Public class Main {
    public static void main (string[]){
        String filename=null;
        if (filename.contains("aaaa")){
            //TODO code application
        }else{
            //TODO code application
        }
    }
```

java.lang.NullPointerException

Implicit vs Explicit

Implicit	Explicit
RunTimeExceptions	Not RunTimeExceptions
May/Should be controlled	Must be controlled
Can't be worked around in run time	May be worked around in run time

`NullPointerException`

unchecked

Implicit vs Explicit

Implicit	Explicit
RunTimeExceptions	Not RunTimeExceptions
May/Should be controlled	Must be controlled
Can't be worked around in run time	May be worked around in run time

`NullPointerException`

unchecked

Explicit Exception

Example

```
Public class Main {
    public static void main (string[]){
        //TODO code application
        String filename="employees.dat";
        FileReader fr = new FileReader(filename);
    }
}
```

Explicit Exception

Example

```
Public class Main {
    public static void main (string[]){
        //TODO code application
        String filename="employees.dat";
        FileReader fr = new FileReader(filename);
    }
}
```

unreported exception FileNotFoundException must be caught or declared to be thorwn

Implicit vs Explicit

Implicit	Explicit
RunTimeExceptions	Not RunTimeExceptions
May/Should be controlled	Must be controlled
Can't be worked around in run time	May be worked around in run time
`NullPointerException`	`FileNotFoundException`
unchecked	checked

“El hombre que ha cometido un error y no lo corrige comete otro error mayor.“

Confucio Filósofo chino