Exception Handling in Java

Jaanus Pöial, PhD

Tallinn, Estonia

Estonian Information Technology College

Errors and Exceptions

Error handling without dedicated tools: return codes, global error states etc.

Problem: it is not reasonable (or even possible) to handle each unusual situation in the same place (subroutine) it occurs. How to separate error handling from the normal control flow?

Estonian Information Technology College

Errors and Exceptions

In Java:

- try / catch (control statement)
- Throwable (specialized objects)
  - Error (program cannot continue)
  - Exception (unusual situation)
    - RuntimeException (no obligation to catch)
- throw (raise exception)
- throws (method heading - delegating)

Errors

- Error
  - LinkageError
    - ClassCircularityError
    - ClassFormatError
    - IncompatibleClassChangeError
      - NoSuchMethodError
      - NoSuchFieldError
      - InstantiationError
      - AbstractMethodError
      - IllegalAccessError
    - NoClassDefFoundError
    - VerifyError
    - AbstractMethodError
    - ExceptionInInitializationError

Errors

- ThreadDeath
- VirtualMachineError
    - InternalError
    - OutOfMemoryError
    - StackOverflowError
    - UnknownError
- AWTError

Checked Exceptions

- Exception
  - ClassNotFoundException
  - CloneNotSupportedException
  - IllegalAccessException
  - InstantiationException
  - InterruptedException
  - NoSuchMethodException
  - TooManyListenersException
  - ParseException
  - AWTException

IOException

- IOException
  - CharConversionException
  - EOFException
  - FileNotFoundException
  - InterruptedIOException
  - ObjectStreamException
    - InvalidClassException
    - InvalidObjectException
    - NotActiveException
    - NotSerializableException
    - OptionalDataException
    - StreamCorruptedException
    - WriteAbortedException

IOException

- SyncFailedException
- UnsupportedEncodingException
- UTFDataFormatException
- MalformedURLException
- ProtocolException
- SocketException
  - BindException
  - ConnectException
  - NoRouteToHostException
- UnknownHostException
- UnknownServiceException

RuntimeException

- RuntimeException
  - ArithmeticException
  - ArrayStoreException
  - ClassCastException
  - IllegalArgumentException
    - IllegalThreadStateException
    - NumberFormatException
    - FormatException
  - IllegalMonitorStateException
  - IllegalStateException
  - IndexOutOfBoundsException
    - ArrayIndexOutOfBoundsException
    - StringIndexOutOfBoundsException

RuntimeException

NegativeArraySizeException

NullPointerException

SecurityException

EmptyStackException

MissingResourceException

NoSuchElementException

IllegalComponentStateException

Try / Catch

```
try {
        block where exceptions may occur;
}
catch (ExcType_1 variable) {
        trap_1;
}
...
catch (ExcType_n variable) {
        trap_n;
}
finally {
        epilogue;
}
```

Estonian Information
Technology College

Example

```
try {

    FileInputStream p = new FileInputStream ("/etc/passwd");
byte[] sisu = new byte [p.available()];
p.read (sisu);
p.close();
System.out.write (sisu);

} catch (FileNotFoundException e) {

    System.out.println ("File not found " + e);

} catch (IOException e) {

    System.out.println ("Input/Output error " + e);

} catch (Exception e) {

    System.out.println ("Something unusual happened " + e);

} finally {

    System.out.println (" This is finally branch");

} // try
```

Throw Statement

To raise an exception in your program

```
   throw throwableObject;
```

Usually error message is provided

```
  throw new SecurityException
                         ("No permission to read!");
```

All checked (not RuntimeExceptions) exceptions need handling – try/catch or delegation "up" using exception declaration in method heading

Estonian Information Technology College

Throws Declaration

```
public static void pause()

   throws InterruptedException {

   Thread.sleep (1000);

}

public Object nextElement()
   throws java.util.NoSuchElementException {
      if (pointerToNext() == null)
         throw new

             java.util.NoSuchElementException();

      else
         return pointerToNext();

}
```

Corresponding javadoc tag!

Problems

- When extending existing exception class provide both default constructor (with no parameters) and a constructor with String parameter (error message).
- No "resume" – use loop structures to continue execution
- Declare needed variables before try-block, otherwise they are not accessible in traps (catch branches)

Examples

- Chaining – ExceptionUsage.java
- Resume and other things – Apples.java