读取Java文件到byte数组的三种方式

**package** zs;

**import** java.io.BufferedInputStream;

**import** java.io.ByteArrayOutputStream;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.IOException;

**import** java.io.RandomAccessFile;

**import** java.nio.ByteBuffer;

**import** java.nio.MappedByteBuffer;

**import** java.nio.channels.FileChannel;

**import** java.nio.channels.FileChannel.MapMode;

**public** **class** FileUtils {

**public** **byte**[] getContent(String filePath) **throws** IOException {

File file = **new** File(filePath);

**long** fileSize = file.length();

**if** (fileSize > Integer.*MAX\_VALUE*) {

System.*out*.println("file too big...");

**return** **null**;

}

FileInputStream fi = **new** FileInputStream(file);

**byte**[] buffer = **new** **byte**[(**int**) fileSize];

**int** offset = 0;

**int** numRead = 0;

**while** (offset < buffer.length

&& (numRead = fi.read(buffer, offset, buffer.length - offset)) >= 0) {

offset += numRead;

}

// 确保所有数据均被读取

**if** (offset != buffer.length) {

**throw** **new** IOException("Could not completely read file "

+ file.getName());

}

fi.close();

**return** buffer;

}

/\*\*

\* the traditional io way

\*

\* **@param** filename

\* **@return**

\* **@throws** IOException

\*/

**public** **static** **byte**[] toByteArray(String filename) **throws** IOException {

File f = **new** File(filename);

**if** (!f.exists()) {

**throw** **new** FileNotFoundException(filename);

}

ByteArrayOutputStream bos = **new** ByteArrayOutputStream((**int**) f.length());

BufferedInputStream in = **null**;

**try** {

in = **new** BufferedInputStream(**new** FileInputStream(f));

**int** buf\_size = 1024;

**byte**[] buffer = **new** **byte**[buf\_size];

**int** len = 0;

**while** (-1 != (len = in.read(buffer, 0, buf\_size))) {

bos.write(buffer, 0, len);

}

**return** bos.toByteArray();

} **catch** (IOException e) {

e.printStackTrace();

**throw** e;

} **finally** {

**try** {

in.close();

} **catch** (IOException e) {

e.printStackTrace();

}

bos.close();

}

}

/\*\*

\* NIO way

\*

\* **@param** filename

\* **@return**

\* **@throws** IOException

\*/

**public** **static** **byte**[] toByteArray2(String filename) **throws** IOException {

File f = **new** File(filename);

**if** (!f.exists()) {

**throw** **new** FileNotFoundException(filename);

}

FileChannel channel = **null**;

FileInputStream fs = **null**;

**try** {

fs = **new** FileInputStream(f);

channel = fs.getChannel();

ByteBuffer byteBuffer = ByteBuffer.*allocate*((**int**) channel.size());

**while** ((channel.read(byteBuffer)) > 0) {

// do nothing

// System.out.println("reading");

}

**return** byteBuffer.array();

} **catch** (IOException e) {

e.printStackTrace();

**throw** e;

} **finally** {

**try** {

channel.close();

} **catch** (IOException e) {

e.printStackTrace();

}

**try** {

fs.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* Mapped File way MappedByteBuffer 可以在处理大文件时，提升性能

\*

\* **@param** filename

\* **@return**

\* **@throws** IOException

\*/

**public** **static** **byte**[] toByteArray3(String filename) **throws** IOException {

FileChannel fc = **null**;

**try** {

fc = **new** RandomAccessFile(filename, "r").getChannel();

MappedByteBuffer byteBuffer = fc.map(MapMode.*READ\_ONLY*, 0,

fc.size()).load();

System.*out*.println(byteBuffer.isLoaded());

**byte**[] result = **new** **byte**[(**int**) fc.size()];

**if** (byteBuffer.remaining() > 0) {

// System.out.println("remain");

byteBuffer.get(result, 0, byteBuffer.remaining());

}

**return** result;

} **catch** (IOException e) {

e.printStackTrace();

**throw** e;

} **finally** {

**try** {

fc.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

}