Class java.io.RandomAccessFile

```
java.lang.Object
   |
   +----java.io.RandomAccessFile
```

---

public class **RandomAccessFile**
extends Object
implements DataOutput, DataInput

Random access files can be constructed from file descriptors, file names, or file objects. This class provides a sense of security by offering methods that allow specified mode accesses of read−only or read−write to files.

Constructor Index

o **RandomAccessFile**(String, String)
Creates a RandomAccessFile with the specified system dependent file name and the specified mode.
o **RandomAccessFile**(int)
Creates a RandomAccessFile with the specified system dependent file descriptor.
o **RandomAccessFile**(File, String)
Creates a RandomAccessFile from a specified File object and mode ("r" or "rw").

Method Index

o **close**()
Closes the file.
o **getFilePointer**()
Returns the current location of the file pointer.
o **length**()
Returns the length of the file.
o **read**()
Reads a byte of data.
o **read**(byte[], int, int)
Reads a sub array as a sequence of bytes.
o **read**(byte[])
Reads data into an array of bytes.
o **readBoolean**()

Reads a boolean.

- **readByte**()
  Reads a byte.
- **readChar**()
  Reads a 16 bit char.
- **readDouble**()
  Reads a 64 bit double.
- **readFloat**()
  Reads a 32 bit float.
- **readFully**(byte[])
  Reads bytes, blocking until all bytes are read.
- **readFully**(byte[], int, int)
  Reads bytes, blocking until all bytes are read.
- **readInt**()
  Reads a 32 bit int.
- **readLine**()
  Reads a line terminated by a '\n' or EOF.
- **readLong**()
  Reads a 64 bit long.
- **readShort**()
  Reads 16 bit short.
- **readUTF**()
  Reads a UTF formatted String.
- **readUnsignedByte**()
  Reads an unsigned 8 bit byte.
- **readUnsignedShort**()
  Reads 16 bit short.
- **seek**(long)
  Sets the file pointer to the specified absolute position.
- **skipBytes**(int)
- **write**(int)
  Writes a byte of data.
- **write**(byte[])
  Writes an array of bytes.
- **write**(byte[], int, int)
  Wrotes a sub array of bytes.
- **writeBoolean**(boolean)
  Writes a boolean.
- **writeByte**(int)
  Writes a byte.
- **writeBytes**(String)
  Writes a String as a sequence of bytes.
- **writeChar**(int)
  Writes a character.
- **writeChars**(String)
  Writes a String as a sequence of chars.
- **writeDouble**(double)
- **writeFloat**(float)
- **writeInt**(int)

Writes an integer.

- **writeLong**(long)
  Writes a long.
- **writeShort**(int)
  Writes a short.
- **writeUTF**(String)
  Writes a String in UTF format.

Constructors

- **RandomAccessFile**

```
public RandomAccessFile(String name,
                        String mode) throws IOException
```

Creates a RandomAccessFile with the specified system dependent file name and the specified mode. Mode "r" is for read−only and mode "rw" is for read+write.

Parameters:

name − the system dependent file name
mode − the access mode

Throws: IOException

If an I/O error has occurred.

- **RandomAccessFile**

```
public RandomAccessFile(int fd) throws IOException
```

Creates a RandomAccessFile with the specified system dependent file descriptor.

Parameters:

fd − the system dependent file descriptor

Throws: IOException

If an I/O error has occurred.

- **RandomAccessFile**

```
public RandomAccessFile(File file,
                        String mode) throws IOException
```

Creates a RandomAccessFile from a specified File object and mode ("r" or "rw").

Parameters:

file − the file object
mode − the access mode

Methods

- **read**

```
public int read() throws IOException
```

Reads a byte of data. This method will block if no input is available.

Returns:

the byte read, or −1 if the end of the stream is reached.

Throws: IOException

If an I/O error has occurred.

o **read**

```
public int read(byte b[],
                int off,
                int len) throws IOException
```

Reads a sub array as a sequence of bytes.

Parameters:

b − the data to be written

off − the start offset in the data

len − the number of bytes that are written

Throws: IOException

If an I/O error has occurred.

o **read**

```
public int read(byte b[]) throws IOException
```

Reads data into an array of bytes. This method blocks until some input is available.

Returns:

the actual number of bytes read, −1 is returned when the end of the stream is reached.

Throws: IOException

If an I/O error has occurred.

o **readFully**

```
public final void readFully(byte b[]) throws IOException
```

Reads bytes, blocking until all bytes are read.

Parameters:

b − the buffer into which the data is read

Returns:

the actual number of bytes read, −1 is returned when the end of the stream is reached.

Throws: IOException

If an I/O error has occurred.

o **readFully**

```
public final void readFully(byte b[],
                            int off,
                            int len) throws IOException
```

Reads bytes, blocking until all bytes are read.

Parameters:

b – the buffer into which the data is read
off – the start offset of the data
len – the maximum number of bytes read

Returns:

the actual number of bytes read, –1 is returned when the end of the stream is reached.

Throws: IOException

If an I/O error has occurred.

o **skipBytes**

```
public int skipBytes(int n) throws IOException
```

o **write**

```
public void write(int b) throws IOException
```

Writes a byte of data. This method will block until the byte is actually written.

Parameters:

b – the byte to be written

Throws: IOException

If an I/O error has occurred.

o **write**

```
public void write(byte b[]) throws IOException
```

Writes an array of bytes. Will block until the bytes are actually written.

Parameters:

b – the data to be written

Throws: IOException

If an I/O error has occurred.

o **write**

```
public void write(byte b[],
                  int off,
                  int len) throws IOException
```

Wrotes a sub array of bytes.

Parameters:

b – the data to be written
off – the start offset in the data
len – the number of bytes that are written

Throws: IOException

If an I/O error has occurred.

o **getFilePointer**

```
public long getFilePointer() throws IOException
```

Returns the current location of the file pointer.

o **seek**

```
public void seek(long pos) throws IOException
```

Sets the file pointer to the specified absolute position.
Parameters:
pos – the absolute position

o **length**

```
public long length() throws IOException
```

Returns the length of the file.

o **close**

```
public void close() throws IOException
```

Closes the file.
Throws: IOException
If an I/O error has occurred.

o **readBoolean**

```
public final boolean readBoolean() throws IOException
```

Reads a boolean.

o **readByte**

```
public final byte readByte() throws IOException
```

Reads a byte.

o **readUnsignedByte**

```
public final int readUnsignedByte() throws IOException
```

Reads an unsigned 8 bit byte.
Returns:
the 8 bit byte read.

o **readShort**

```
public final short readShort() throws IOException
```

Reads 16 bit short.

Returns:
the read 16 bit short.

o **readUnsignedShort**

```
public final int readUnsignedShort() throws IOException
```

Reads 16 bit short.
Returns:
the read 16 bit short.

o **readChar**

```
public final char readChar() throws IOException
```

Reads a 16 bit char.
Returns:
the read 16 bit char.

o **readInt**

```
public final int readInt() throws IOException
```

Reads a 32 bit int.
Returns:
the read 32 bit integer.

o **readLong**

```
public final long readLong() throws IOException
```

Reads a 64 bit long.
Returns:
the read 64 bit long.

o **readFloat**

```
public final float readFloat() throws IOException
```

Reads a 32 bit float.
Returns:
the read 32 bit float.

o **readDouble**

```
public final double readDouble() throws IOException
```

Reads a 64 bit double.
Returns:

the read 64 bit double.

o **readLine**

```
public final String readLine() throws IOException
```

Reads a line terminated by a '\n' or EOF.

o **readUTF**

```
public final String readUTF() throws IOException
```

Reads a UTF formatted String.

o **writeBoolean**

```
public final void writeBoolean(boolean v) throws IOException
```

Writes a boolean.
Parameters:
v – the boolean value

o **writeByte**

```
public final void writeByte(int v) throws IOException
```

Writes a byte.
Parameters:
v – the byte

o **writeShort**

```
public final void writeShort(int v) throws IOException
```

Writes a short.
Parameters:
v – the short

o **writeChar**

```
public final void writeChar(int v) throws IOException
```

Writes a character.
Parameters:
v – the char

o **writeInt**

```
public final void writeInt(int v) throws IOException
```

Writes an integer.
Parameters:
v – the integer

o **writeLong**

```
public final void writeLong(long v) throws IOException
```

Writes a long.
Parameters:
v – the long

o **writeFloat**

```
public final void writeFloat(float v) throws IOException
```

o **writeDouble**

```
public final void writeDouble(double v) throws IOException
```

o **writeBytes**

```
public final void writeBytes(String s) throws IOException
```

Writes a String as a sequence of bytes.
Parameters:
s – the String

o **writeChars**

```
public final void writeChars(String s) throws IOException
```

Writes a String as a sequence of chars.
Parameters:
s – the String

o **writeUTF**

```
public final void writeUTF(String str) throws IOException
```

Writes a String in UTF format.
Parameters:
str – the String