

RandomAccessFile и т.д.

Java Collections
2 уровень, 1 лекция

ОТКРЫТА

- Привет, Амиго!
- Привет, Билаабо! Как жизнь?
- Отлично. Вчера выводил паразитов, но пока не очень-то получается. А потом опять пришлось ночевать в мусорном баке.
- Т.е. все по-прежнему отлично?
- Можно и так сказать.
- Гуд. А что у нас будет сегодня?
- Сегодня я тебе расскажу про класс **RandomAccessFile**.



Дело в том, что `FileInputStream` и `FileOutputStream` представляют файлы в виде потоков: читать из них и писать в них можно только последовательно.

Это не всегда очень-то удобно. Иногда тебе нужно записать пару строк в середину файла, или прочитать пару страниц текста в конце многомегабайтного файла. Читать для этого весь файл не очень эффективно.

Для решения этой проблемы был создан класс **RandomAccessFile**. С его помощью можно писать в любое место файла, читать из него, а также писать и читать файл одновременно.

- Как интересно.
- Ага. Очень удобно на самом деле.
- А как читать из произвольного места?

— Все довольно просто. Представь, что у тебя открыт текстовый редактор «блокнот». В нем есть курсор. Когда ты что-то печатаешь, текст добавляется в том месте, где он стоит. С чтением файла то же самое. Чтение происходит в том месте, где стоит «курсор». При чтении/записи он сам автоматически сдвигается.

Давай, я лучше покажу тебе пример:

Чтение файла:	
1	<code>//r-</code> read, файл открыт только для чтения
2	<code>RandomAccessFile raf = new RandomAccessFile("input.txt", "r");</code>

НАЧАТЬ ОБУЧЕНИЕ

```
5 raf.seek(100);
6
7 //читаем строку, начиная с текущего положения курсора и до конца строки
8 String text = raf.readLine();
9
10 //закрываем файл
11 raf.close();
```

В этом примере я хотел бы обратить твоё внимание на две вещи:

Во-первых, создание объекта **RandomAccessFile**. Вторым параметром идёт буква **r**. Это означает, что файл открыт для чтения (**r**- **read**). Если ты хочешь открыть файл для чтения и записи, в конструктор надо передать “**rw**” вместо “**r**”.

Во-вторых, обрати внимание на метод **seek**. С помощью этого метода можно прыгать по файлу и менять позицию курсора для текущей операции чтения/записи. Сразу при открытии файла «курсор» устанавливается на 0-й байт. Или точнее – перед нулевым байтом.

— Правильно ли я понял, что мы открыли файл, и курсор был в его самом начале – на позиции 0. Затем мы вызвали **seek** и он переместился на 100-й байт. А когда вызвали **readLine**, то чтение уже было начиная с сотого байта. Так?

— Да. Только хочу обратить твоё внимание на то, что метод **seek** позволяет произвольно прыгать по файлу. Пример:

Чтение файла:

```
1 //r- read, файл открыт только для чтения
2 RandomAccessFile raf = new RandomAccessFile("input.txt", "r");
3 // «курсор» стоит на 0-м символе.
4 String text1 = raf.readLine();
5
6 //перемещаем «курсор» на 100-й символ.
7 raf.seek(100);
8 String text2 = raf.readLine();
9
10 //перемещаем «курсор» на 0-й символ.
11 raf.seek(0);
12 String text3 = raf.readLine();
13
14 //закрываем файл
15 raf.close();
```

В данном примере мы вначале прочитали строку, начиная с 0-го байта. Затем прыгнули на сотый байт и прочитали строку там. Затем снова прыгнули на 0-й байт и прочитали строку. Т.е. **text1** и **text3** – это идентичные строки.

— Ага. Ситуация начинает проясняться.

— Отлично. Тогда вот тебе ещё один пример:

Чтение файла:

```
1 //rw- read/write, файл открыт и для чтения и для записи
2 RandomAccessFile raf = new RandomAccessFile("seek.txt", "rw");
3
4 //пишем в файл строку, начиная с 0-го байта
5 raf.writeBytes("It is a string");
6
7 //ставим курсор на 8-й символ
8 raf.seek(8);
```

```
10 raf.writeBytes("surprise!");
11
12 //закрываем файл
13 raf.close();
```

Тут мы открываем файл для чтения и записи – в конструктор передаем «rw» (read/write).

Затем пишем в файл строку «It is a string».

Затем переставляем курсор на 8-й байт (как раз на начало слова string)

Затем пишем в файл строку «surprise!»

В результате файл будет содержать «It is a surprise!»

— Т.е. байты не вставляются в середину файла, а заменяют те, которые там были?

— Ага.

— А если мы установим курсор в самый конец файла?

— Тогда байты будут писаться в конец, а файл – удлиняться. Т.е. практически то же самое, когда ты пишешь текст в текстовом редакторе.

— Гм. Вроде все понятно. А можно полный список методов класса **RandomAccessFile**?

— Конечно. Держи:

Метод	Описание
<code>int read()</code>	Читает один байт и возвращает его
<code>int read(byte b[], int off, int len)</code>	Читает массив байт
<code>int read(byte b[])</code>	Читает массив байт
<code>void readFully(byte b[])</code>	Читает массив байт, ждет, пока добавятся новые байты, если их не хватает для заполнения массива
<code>int skipBytes(int n)</code>	Пропускает n байт. Т.е. перемещает курсор на n байт вперед
<code>void write(int b)</code>	Пишет один байт в то место, где стоит курсор
<code>void write(byte b[])</code>	Пишет массив байт в то место, где стоит курсор
<code>void write(byte b[], int off, int len)</code>	Пишет массив байт в то место, где стоит курсор
<code>long getFilePointer()</code>	Возвращает номер байта, на который указывает «курсор». Может быть от 0 до «длины файла»
<code>void seek(long pos)</code>	Перемещает «курсор», используемый для чтения/записи, в указанное место
<code>long length()</code>	Возвращает длину файла
<code>void setLength(long newLength)</code>	Устанавливает новую длину файла. Если файл был больше – он обрезается, если меньше – расширяется и новое место заполняется нулями

Andrew Karelin

Уровень 41, Sumy, Украина

21 апреля, 17:45

...

Интересно, почему разрабы пишушие подобные классы не вставляют в них, например, возможность вставки байтов (текста) в файл с автоматической возможность сдвига. Ну разу же написали клас...доделали б уже до финала....или это чтобы жизнь "малиной" не казалась?

Ответить

+1

...

Vladimir

Уровень 35, Россия

23 апреля, 16:29

...

Соглашусь, есть много вопросов к данному классу
Возможно они посчитали, что это был бы довольно затратный метод, т.к. по факту при добавлении нового байта пришлось бы перемещать всю оставшуюся часть файла на один байт вперед (при множественной записи думаю, что это накладно).
Вообще, как по мне, смысл данного класса скорее в считывании проиндексированных данных или данных с рассчитываемой позицией нужных байт. Хотя почему они не добавили нормальное считывание строк с UTF-8 для меня загадка.

Ответить

+1

...

Boston

Уровень 26, Russian Federation

19 марта, 20:50

...

Чего то не понял:
//ставим курсор на 8-й символ
raf.seek(8);
...
Затем переставляем курсор на 8-й байт (как раз на начало слова string)
8 байт или символов всё таки? На что указывает курсор?

Ответить

+1

...

StringiD

Престарелый студент в Жавараш

28 марта, 11:14

...

Так нужно уже привыкнуть к методу обучения взорви мозг. 1 байт = 1 символ судя по всему.

Ответить

0

...

Andrew Karelin

Уровень 41, Sumy, Украина

21 апреля, 17:36

...

а в кириллице - 1 символ 2 байта.
Так едем по символам или байтам...непонятно.

Ответить

+1

...

Alexander Tsaregorodtsev

Уровень 35, Новосибирск, Russian Federation

20 мая, 06:26

...

В примере текст только из латинских букв состоит, а каждая латинская буква одним байтом кодируется. Так что в данном конкретном случае разницы нет. Если будет русский текст, то в нём уже каждая буква двумя байтами будет кодироваться.

Ответить

0

...

Andrew Karelin

Уровень 41, Sumy, Украина

20 мая, 23:46

...

Вопрос не про "данный пример", а про то, в чём считает метод seek(8)?
В символах, или в байтах?
В смысле, если я открываю файл с кириллицей, то seek(8)мне сместит курсор на 8, или на 4 символ...вот в чем вопрос.
Ясности нет.

Ответить

0

...

Dmitriy Shebryuk

Уровень 23, Москва, Russian Federation

19 июня, 07:36

...

В байтах.

Ответить

0

...

LuneFox

инженер по сопровождению в BIFIT

EXPERT

21 декабря 2021, 00:18

...

А как написать в середину файла со сдвигом всего написанного вправо? Считать остаток в массив байт, написать что надо, а потом дополнить сохранённым массивом? А если там 100500 МБ данных?

Ответить

0

...

Roman Konchalo

Уровень 28, Россия

EXPERT

20 января, 10:39

...

А. Сделать пару буфферов по размеру данных, которые собираемся записать: 1 для того что собираемся вставлять, 2 для того участка, который надо сдвинуть. Из того участка, куда вставляем, сохраняем во второй буффер все данные, в "освободившийся" участок вставляем необходимое из первого буффера. На следующем чётном шаге буферы меняются местами, следующий участок данных, которые мы будем заменять, будет браться уже из 2 буффера, а 1 будет служить для сохранения перезаписываемого участка. И так действие повторяется до конца файла.
Б. Взять один буффер фиксированного размера (4096 байт например) и удлинить файл на размер вставляемых данных, а потом перемещать данные по блокам буффера из старого положения в конец, пока не будет достигнут тот участок, с которого будут записаны новые данные, освободив для него место таким образом
Для буффера будет удобно использовать ByteBuffer, особенно в случае А

Ответить

0

...

LuneFox

инженер по сопровождению в BIFIT

EXPERT

20 января, 11:41

...

То есть. если я хочу сдвинуть 10ГБ данных. мне нужно лепать буффер на 10ГБ?

Нужно использовать вариант Б. Или он не подходит уже?)

Ответить

0

LuneFox

инженер по сопровождению в BIFIT

EXPERT

20 января, 20:39

Ну, как я понял, в варианте Б придётся перезаписывать всё те же 10ГБ, только блоками по 4КБ?

Ответить

0

Роман Кончалов

Уровень 28, Россия

EXPERT

20 января, 20:50

Перезаписывать придётся размер от точки начала записи до конца файла, ну это уже арифметика начального класса, даже не хотел отвечать сначала на этот вопрос. Да любого размера. Почему 4Кб? Это был пример из рекомендации. Можно взять размер данных от точки записи до конца файла и за один присест перенести. Можно взять размер самого маленького кэша целевых устройств и надеяться, что так быстрее перенесётся, а ОС не порубит этот кэш на маленькие куски, удлиннив время ожидания на системные прерывания

Ответить

0

LuneFox

инженер по сопровождению в BIFIT

EXPERT

21 января, 12:21

Так я про то и говорю. Скажем, у нас файл на 14 ГБ, и нам нужно записать пару байтиков в место, расположенное через 4 ГБ от начала файла. Получается, чтобы вставить туда эти два байтика мы должны будем двигать остальные 10 ГБ.

Ответить

0

Rock133

Java Intern

13 апреля 2021, 17:07

Билабо хороший чел. Вот только почему у них доктор в бачке ночует? И не надо мне тут про паразитов)

Ответить

0

Flexo

Bending Unit #3370318

14 марта 2021, 21:17

хм, методы write() пишут только байты, а writeBytes пишет строки (и иногда байты).
верх логики

Ответить

+2

Лозко Ростислав Владимирович

Уровень 41

8 июля 2021, 10:57

Для уточнения writeByte пишет байты, а writeBytes пишет строки.

Ответить

+1

Роман Кончалов

Уровень 28, Россия

EXPERT

20 января, 11:01

Возможно это связано с постоянными перестановками в классе String, там символы то в массиве байтов хранились, то в массиве char с переменными offset и count, а сейчас опять в байтах и кодере.
upd.: в популярных комментариях уже разжевали всё, не ленитесь сортировать их соответствующей кнопкой в начале)

Ответить

0

Филипп

Уровень 27, Россия

21 января 2021, 01:57

Как всегда подробно и основательно 😊
Почитал в [документации Оракулов](#), такое впечатление, что класс создан только для работы с **четко размеченными данными**. Он основывается на интерфейсах DataInput и DataOutput, и классы реализующие их предназначены для работы с изображениями, объектами и кэшем памяти.

Особо стоит отметить 4 метода для работы со строками: [readLine\(\)](#), [writeBytes\(String s\)](#), и "*не замеченные*" [readUTF\(\)](#) и [writeUTF\(String str\)](#).
Первые 2 пишут/читают обрезая старшие биты в 0, таким образом они не поддерживают стандарт UTF. вторая пара еще веселее они кодируют полную последовательность UTF, но на свой манер([Modified UTF-8](#)), где 1 символ соответствует 1-3 байтам, таким образом совершенно руша совместимость с любыми другими программами. Что еще важно, вторая пара в начале пишет 2 байта для обозначения длины записанной строки, и чтец. соответственно эти 2 байта читает и пытается забрать строку именно такой длины, никаких /r /n.

Метод [seek\(long pos\)](#) может быть установлен вообще на любую позицию, но размер файла увеличится только если начать что-то в эту позицию писать, в отличии от [setLength\(long newLength\)](#).

Так же все остальные данные строго структурированы, например: [writeFloat\(float v\)](#).- сначала конвертирует число с плавающей запятой в целое, используя [floatToIntBits](#) и уже потом пишет. так же не указанные здесь [int readUnsignedByte\(\)](#) и [int readUnsignedShort\(\)](#) - предназначенные для считывания беззнаковых чисел (хотя таких типов в Яве и нет, но для данных это может быть принципиально)
// тудыть их растудыть, они что, спецом не указали самые отличительные методы по сравнению с обычными классами для чтения/записи? Не ну должен же быть какой-то предел...

Ответить

+21

VDT

Java Developer в USETECH

25 марта 2021, 16:26

Я тэбэ умоляю.... Похоже человек сдавший Раж и решивший все 100% задач ... сори... сдавший их валидатору... Считай мидл сразу. Видимо такой принцип данного обучения.

НАЧАТЬ ОБУЧЕНИЕ

Vladimir “Rain_Senpai1995” Soldatenko

Уровень 35, Киев, Украина

16 января 2021, 18:21

...

После прочтения "сиего", приблизительно понял как работать с RandomAccessFile. Хотелось бы больше конкретики - где это всё применять в каких случаях это всё применять?

Ответить

0

0

Duckpool

утка в в бассейне

2 июля 2020, 00:32

...

как это относится к коллекциям? 🤔

Ответить

+3

+3

DZ

Уровень 35, Россия

8 декабря 2020, 23:04

...

Первые вроде пять или шесть уровней в соседнем курсе мы задавали похожий вопрос: и где же тут треды? Я так подозреваю, что тут тоже будет скоро большая задача с коллекциями, которая будет использовать все предыдущие уровни

Ответить

+3

+3

Kex

Уровень 38, Тольятти

EXPERT

26 июня 2020, 10:55

...

Дикая хрень или пора начать разбираться в кодировках(После статьи одни вопросы, полез в коменты вниз, так вопросов стало в 10 раз больше и смотрю у меня не у одного так.

Ответить

+2

+2

Herr Ives

Уровень 30

21 января 2021, 17:00

...

посмотрите лекции головача по I/O там начинается как раз с кодировок

Ответить

+1

+1

KsArT

Уровень 41, Харьков, Украина

22 июня 2020, 08:53

...

Следует отметить, что если мы пишем в файл по индексу, где данные уже присутствуют, то они будут заменены.

Ответить

+2

+2

↺

Показать еще комментарии

ОБУЧЕНИЕ

Курсы программирования

Курс Java

Помощь по задачам

Подписки

Задачи-игры

СООБЩЕСТВО

Пользователи

Статьи

Форум

Чат

Истории успеха

Активности

КОМПАНИЯ

О нас

Контакты

Отзывы

FAQ

Поддержка

JavaRush — это интерактивный онлайн-курс по изучению Java-программирования с нуля. Он содержит 1200 практических задач с проверкой решения в один клик, необходимый минимум теории по основам Java и мотивирующие фишки, которые помогут пройти курс до конца: игры, опросы, интересные проекты и статьи об эффективном обучении и карьере Java-девелопера.

ПОДПИСЫВАЙТЕСЬ

ЯЗЫК ИНТЕРФЕЙСА

Русский

НАЧАТЬ ОБУЧЕНИЕ



"Программистами не рождаются" © 2022 JavaRush