



Потоки ввода/вывода

Java Core (/quests/QUEST_JAVA_CORE)
8 уровень (/quests/lectures/?quest=QUEST_JAVA_CORE&level=8), 1 лекция (/quests/lectures/questcore.level08.lecture01)

ОТКРЫТА

— Привет, Амиго! Сегодня мы будем знакомиться с потоками ввода-вывода. Пару дней назад мы немного цепляли данную тему, но сегодня пройдемся по ней основательно. Потоки ввода-вывода делятся на 4 категории:

- 1) Потоки делятся по направлению: потоки ввода и потоки вывода
- 2) Потоки делятся по типу данных: работают с байтами или работают с символами.

Таблица:

Поток ввода
Поток вывода
Работает с байтами
InputStream
OutputStream
Работает с символами
Reader
Writer

Если объект реализует интерфейс InputStream, значит, он поддерживает последовательное чтение из него байт (byte).

Если объект реализует интерфейс OutputStream, значит, он поддерживает последовательную запись в него байт (byte).

Если объект реализует интерфейс Reader, значит, он поддерживает последовательное чтение из него символов (char).

Если объект реализует интерфейс Writer, значит, он поддерживает последовательную запись в него символов (char).



Поток вывода напоминает принтер. На принтер мы можем выводить документы. В поток вывода мы можем выводить данные.

Тогда поток ввода можно сравнить со сканером, ну или с розеткой. С помощью сканера мы можем ввести документы к себе в компьютер. Также мы можем подключиться к розетке и получать из нее электричество. Из потока ввода мы можем получать данные.

— А где они используются?

— Эти классы используются в Java повсеместно. Известный нам `System.in` – это статическая переменная по имени `in` типа `InputStream` в классе `System`.

— Надо же! Оказывается, все это время я пользовался потоком `InputStream` и не знал об этом. `System.out` – тоже поток?

— Да, `System.out` – это статическая переменная по имени `out` типа `PrintStream` (наследник `OutputStream`) в классе `System`.

— Т.е. я все время пользовался потоками и даже не подозревал об этом?

— Да, и это говорит лишь о том, насколько такие потоки удобны. Просто берешь и пользуешься.

— Хотя этого нельзя сказать про `System.in`. К нему постоянно приходилось добавлять `BufferedReader` и `InputStreamReader`.

— Да, это так. Но на это тоже были свои причины.

Видишь ли, типов данных очень много, как и способов работы с ними. Поэтому количество стандартных классов ввода-вывода очень быстро росло, хоть они и делали все почти то же самое. Чтобы избежать такой сложности, разработчики Java применили принцип абстракции и разделили классы на много маленьких частей.

Зато их можно соединить последовательно и получить очень сложную функциональность, если она тебе понадобилась. Смотри пример:

Вывод строки на консоль

```
1 System.out.println("Hello");
```

Сохранили поток вывода на консоль в отдельную переменную.

Выводим в поток строку.

```
1 PrintStream console = System.out;  
2 console.println("Hello");
```

Создали динамический (растягивающийся) массив байт в памяти.

Связали его с новым потоком вывода – объектов `PrintStream`

Выводим в поток строку.

```
1 ByteArrayOutputStream stream = new ByteArrayOutputStream();  
2 PrintStream console = new PrintStream(stream);  
3 console.println("Hello");
```

— Действительно, чем-то похоже на конструктор `Lego`. Только непонятно, что весь этот код делает.

— Пусть это тебя не беспокоит сейчас. Всему свое время.

Хочу, чтобы ты запомнил вот что: если класс реализует интерфейс `OutputStream` – он позволяет записывать в него байты. Почти так же, как ты выводил данные на консоль. Что он будет с этими данными делать – его задача. В «конструкторе» важно не назначение отдельного элемента, а насколько классные вещи мы можем собрать, благодаря многообразию существующих элементов.

— Хорошо. Тогда с чего мы начнем?

[< \(/quests/lectures/questcore.level08.lecture00\)](/quests/lectures/questcore.level08.lecture00)

[×18 > \(/quests/lectures/questcore.level08.lecture02\)](/quests/lectures/questcore.level08.lecture02)

Комментарии (20)


популярные

новые

старые

Никита

Assanali 19 уровень

22 марта, 05:35 

думал сейчас будет задание по набору кода.
но нет.
что-то давно их не было)

Ответить

0

Инга-Виктория Куксова 19 уровень, Санкт-Петербург

30 января, 21:07 

— Действительно, чем-то похоже на конструктор Lego. Только непонятно, что весь этот код делает.


— Пусть это тебя не беспокоит сейчас. Всему свое время.

Неужели авторы текста думают, что если я ничего не пойму, то запомню?? Местами теория хромает на обе ноги.

Ответить

+3

Redas Shuliakas 21 уровень

9 февраля, 10:51 

Скоро твоя надежда растает, надежда о том что теория каким то образом тебе поможет в написании чего либо, то есть конечно поможет, но не настолько насколько ты думаешь)

Ответить

+4

Вадим Чубаров 18 уровень

24 февраля, 16:46 

Вообще пока не пойму зачем теория)))

Ответить

+1

Harvey 27 уровень, Минск

5 октября 2017, 23:01 

Господи, пару дней назад мы цепляли тему потоков ввода-вывода?
У меня уже вечность прошла

Ответить

+35

Egorro 40 уровень

28 сентября 2017, 14:10 

Хочу, чтобы ты запомнил вот что: если класс реализует интерфейс OutputStream – он позволяет записывать в него байты.

Только вот в примере парой строчек выше мы записываем в этот класс не именно байты, а последовательность символов, а символы, как и любые другие типы данных, по сути, и являются наборами байт. Соответственно, что именно в данном случае нужно запомнить - в упор не понятно...

Ответить

+2

Donatello 32 уровень

6 сентября 2017, 02:15 

Ого, заслужили лекцию по потокам, наконец-то)

Ответить

+6

albaslug 36 уровень, Санкт-Петербург

16 августа 2017, 04:31 

"...реализует интерфейс InputStream..."

Но ведь все перечисленное - не интерфейсы, а абстрактные классы:

```
public abstract class OutputStream implements Closeable, Flushable
```

и т. п.

Ответить

+8

soloyes 28 уровень


18 октября 2017, 20:04 

В книге Экель объясняет, что при наследовании мы обходимся с наследниками которые реализуют чей-то интерфейс. Не обязательно должно быть ключевое слово interface. В конце концов, interface это абстрактный класс, у которого все методы абстрактные.

Ответить

+1

Инга-Виктория Куксова 19 уровень, Санкт-Петербург

30 января, 21:10 

Как я поняла, самая большая разница интерфейса и абстрактного класса в том, что наследовать можно только один класс, и это очень большой минус, а реализовывать сколько угодно

только только один класс, и это очень хороший минус, а реализовывать несколько удобных интерфейсов, а так больше нет особой разницы.

Ответить

0

MaxLich 40 уровень, Санкт-Петербург

28 марта 2017, 10:40

Как тут кто-то правильно написал: ввод - это ввод в программу (в том числе и чтение из файла), а вывод - это вывод из программы (на консоль, в отдельный файл или на принтер).

Ответить

+8

Vitya 33 уровень, Нижний Новгород

21 июня 2017, 00:03

Лучше представлять центром Оперативную память. Мы либо вводим в неё данные, либо выводим.

Ответить

+3

Джонни Мнемоник 24 уровень

25 августа 2017, 07:47

Ну или поток в памяти. Потоки.

Ответить

0

Александр Лабкович 19 уровень

25 марта 2017, 10:04

Спасибо, Anton Stukov. Не внимательно прочитал, не заметил что в переменную печатаем "Hello" и долго не мог понять как вообще в поток попадет наша строка.

Ответить

0

Anton Stukov 28 уровень

21 марта 2017, 08:50

Он не выводит на консоль "Hello".

Ответить

+1

Джонни Мнемоник 24 уровень

25 августа 2017, 08:04

Мдя... Вывод в поток... Ввод из потока... К этому надо привыкнуть. Моск развернулся на 180°. Да он выводит в поток. А потом уже из потока вводим на консоль с помощью println(); Если , конечно, я правильно понял.

Ответить

0

Valeriy 25 уровень

9 октября 2017, 22:10

Наверное все-таки так (напишу может и сам пойму): println принимает последовательность символов, PrintStream переводит в последовательность байт, ByteArrayOutputStream заполняет массив байт. А зачем? Нет не помогло - не понял :). Может не забивать пока голову, а просто пользоваться?

Ответить

0

ddmxm 21 уровень, Москва

16 марта 2017, 13:50

"Только непонятно, ЧТО весь этот код делает." - это как-раз понятно (выводит на консоль "Hello"). Непонятно может быть КАК он это делает. То есть детали реализации.

Ответить

0

Максим Караваев 40 уровень, Санкт-Петербург

14 сентября 2017, 13:04

В третьем примере "Hello" выводится не на консоль.

Ответить

+2

[jsh.ru/](https://javarush.ru/)) [G+ \(https://plus.google.com/114772402300089087607/\)](https://plus.google.com/114772402300089087607/) [_ \(https://twitter.com/javarush_ru/\)](https://twitter.com/javarush_ru/)



Программистами не рождаются
© 2018