FileReader/FileWriter

Java Core
9 уровень, 4 лекция

ОТКРЫТА

— Привет, Амиго! Вчера Риша рассказывал тебе про **FileInputStream**, **FileOutputStream**. А сегодня я расскажу тебе о классах **FileReader** и **FileWriter**.

Как ты уже, наверное, догадался, все эти классы являются адаптерами между объектом File и “интерфейсами” **InputStream**, **OutputStream**, **Reader**, **Writer**.

— Они похожи на адаптеры между File и Reader/Writer, вот только в конструктор к ним нужно передать объект String, а не File!

— На самом деле, у них несколько конструкторов: есть и File и String. И если ты передашь в конструктор класса объект типа String, то в нем, незаметно для тебя создастся объект типа File, с путем файла, взятым из переданного String.

Это сделано для удобства. Разработчики Java взяли самые частые сценарии использования этих классов, и написали для всех их конструкторы. Это очень удобно, не так ли?

— Да, удобно, согласен. Но почему тогда мне постоянно приходится писать:
BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));
Почему они не добавили этот сценарий?

— Дело в том, что типичная программа на Java не работает с консолью. Вернее, почти никогда ничего с нее не читает. Это может быть web-сервер, сервер приложений или еще какая-нибудь сложная система.

Но для вывода данных и текста на консоль добавили объект PrintStream. Так как те же «серверные программы» часто пишут в консоль свое состояние работы, ошибки и еще разную информацию.

— Понятно. А копировать файл с помощью FileReader и FileWriter тоже можно?

— Да, если он текстовый (т.е. состоит из символов). Вот, смотри пример:

Копируем файл на диске

```
1  public static void main(String[] args) throws Exception
2  {
3   FileReader reader = new FileReader("c:/data.txt");
4   FileWriter writer = new FileWriter("c:/result.txt");
5
6   while (reader.ready()) //пока есть непрочитанные байты в потоке ввода
7   {
8    int data = reader.read(); //читаем один символ (char будет расширен до int)
9    writer.write(data); //пишем один символ (int будет обрезан/сужен до char)
10  }
11
12  //закрываем потоки после использования
13  reader.close();
14  writer.close();
15 }
```

— Почти никаких отличий.

— Да, отличия минимальны.

‹ Предыдущая

×19 ›

0 0 0 0 0

+5

Комментарии (13) популярные новые старые

Программистами не рождаются
© 2018