Java IO

Немного определений

Java IO (Input/Output) предоставляет богатый набор классов и интерфейсов для чтения и записи данных, как из файлов, так и из других источников или приемников данных.

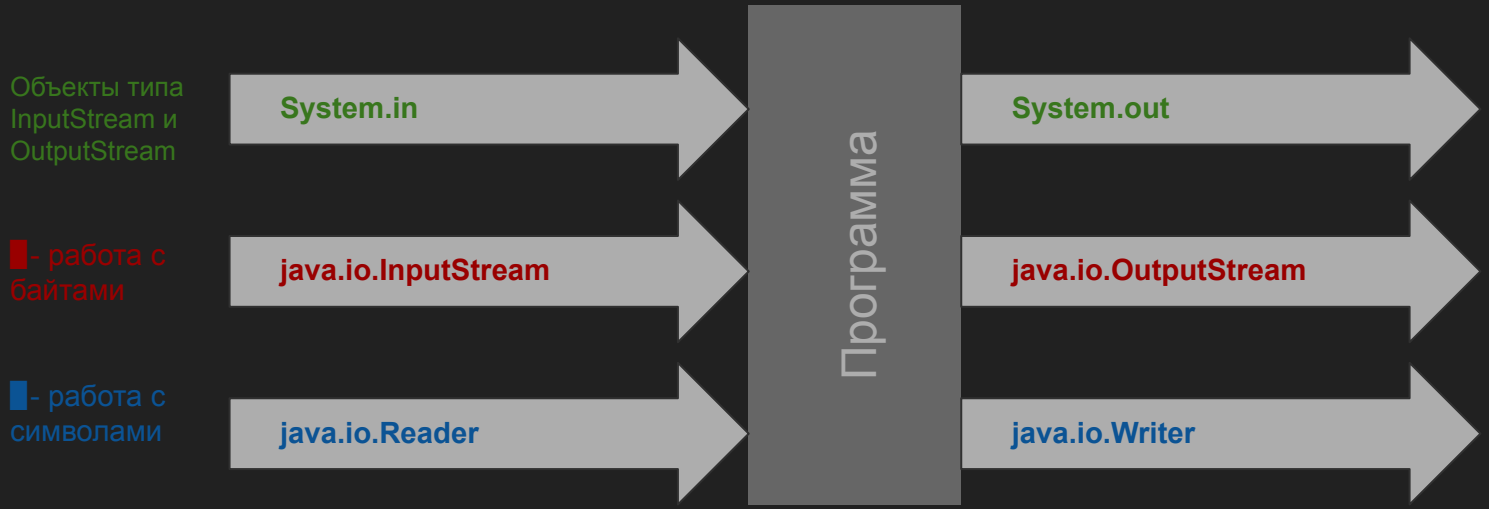
InputStream и OutputStream являются абстрактными базовыми классами для чтения и записи байтовых данных соответственно.

(Random meme for mental health)

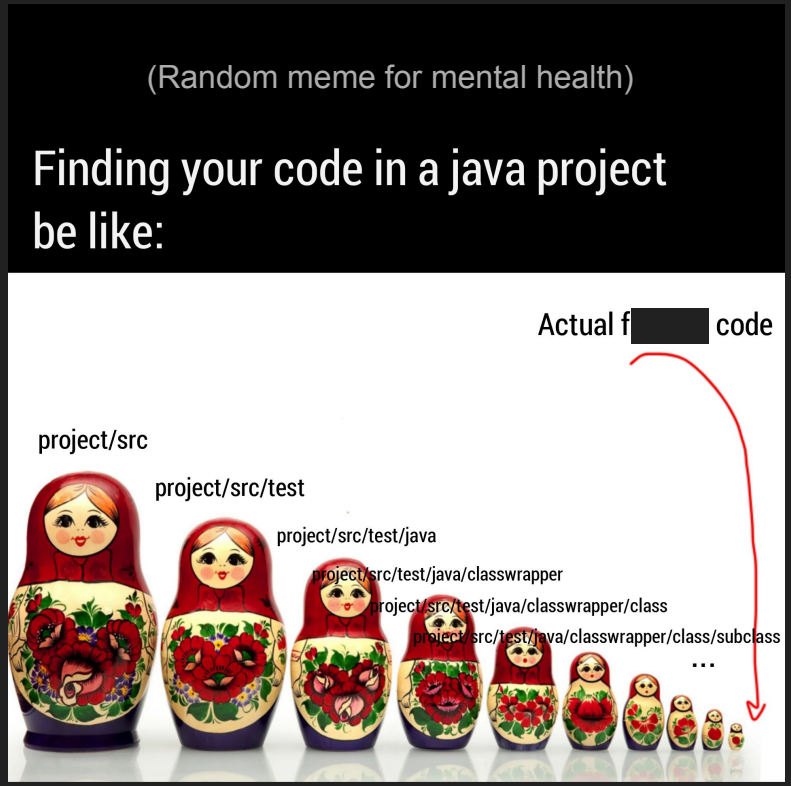
Потоки ввода-вывода

System.in - поток для ввода в консоль
InputStream - класс потока ввода
Reader - класс потока ввода

System.out - поток для вывода в консоль
OutputStream - класс потока вывода
Writer - класс потока вывода



Немного определений



Байтовые и символьные потоки данных (I/O streams)

- Поток — последовательность данных (байтов, символов, примитивных типов, объектов)
- Поток ввода — для чтения данных из источника
- Поток вывода — для записи данных в приемник

Старый интерфейс работы с файлами — класс File

Чтение потока ввода

```java
package l43.slides.ex1;

import java.io.FileInputStream;
import java.io.IOException;
import java.util.Scanner;

// FileInputStream используется для чтения данных из файла.
// Scanner - полезная обертка для работы с потоками ввода
public class WithTryResourcesScanner {
   public static void main(String[] args) {
       try (
           FileInputStream fis =  new
FileInputStream("file.txt")
       ) {
           Scanner scanner =  new Scanner(fis);
           while(scanner.hasNext()) {
               // читаем построчно пока есть данные
               String line = scanner.nextLine();
               System.out.print(line);
           }
       } catch (IOException e) {
           System.out.print("Ошибка ввода вывода");
       }
   }
}
```

1. Открываем файл используя try с ресурсами и получаем InputStream
2. Используем обертку Scanner для работы с InputStream
3. Пока есть новая строка в поток - считываем стредующую строку и выводим
4. Если возникает ошибка при работе с потоком ввода - выводим сообщение о проблеме
5. В любом случае закрываем файл

Альтернатива — использовать InputStream без обертки Scanner: неудобно, но надо понимать что внутри есть (см. примеры кода)

Запись в поток вывода

```java
package l43.slides.ex2;

import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;

// BufferedWriter используется для буферизированной записи
// текста в файл. Метод write() используется для записи
// текста, а newLine() для добавления новой строки.
public class WriteByWriter {
   public static void main(String[] args) {
       try (FileWriter fw = new FileWriter("file.txt")) {
           BufferedWriter bw = new BufferedWriter(fw);
           // Записываем строку в файл
           bw.write("Привет, мир!");
           // Добавляем новую строку
           bw.newLine();
           bw.write("До свидания, мир!");
       } catch (IOException e) {
           System.out.print("Ошибка ввода вывода");
       }
   }
}
```

1. Открываем файл используя try с ресурсами и получаем OutputStream обернутый в FileWriter
2. Используем обертку Buffered для работы с FileWriter
3. Записываем строки
4. Если возникает ошибка при работе с потоком ввода - выводим сообщение о проблеме
5. В любом случае закрываем файл

Альтернатива — использовать OutputStream без оберток FileWriter и BufferedWriter: неудобно, но надо понимать что внутри есть (см. примеры кода)

Выводы

- Использование InputStream и OutputStream подходит для работы с байтовыми данными
- Reader и Writer лучше подходят для работы с текстовыми данными
- Буферизированные версии (BufferedReader и BufferedWriter) обеспечивают высокую производительность за счет использования внутренних буферов

Контрольная точка

- Понятно ли?

Если все ясно, ставим плюсы, иначе - задаем вопросы.

Домашнее задание

- Ознакомиться с документацией и статьей по ссылкам со второго слайда
  - подробно понятным языком: https://metanit.com/java/tutorial/6.8.php (по желанию)
- Кратко
  - см. отдельные разделы по ссылке выше
- Задачки
  - Минимум:
    - записать в файл “Hello world, I’m тут ваше имя” (если его нет, он будет создан)
    - прочитать этот файл и вывести его содержимое
  - Максимум:
    - напишите чтение файла используя Scanner, BufferedReader, FileReader, FileInputStream
    - напишите чтение файла используя BufferedWriter, FileWriter, FileOutputStream

The end