

Gravação de objetos em arquivos

Programação Orientada a Objetos

Prof. Tulio Alberton Ribeiro

Instituto Federal de Santa Catarina – IFSC
campus São José
tulio.alberton@ifsc.edu.br

17 de setembro de 2014



Serialização / DeSerialização

- A chave para armazenar e recuperar a informação contida em objetos Java é através de serialização.



Serialização / DeSerialização

- A chave para armazenar e recuperar a informação contida em objetos Java é através de serialização.
- Serializar a informação, é armazená-la junto com informação suficiente para poder reconstruí-la.



Serialização / DeSerialização

- A chave para armazenar e recuperar a informação contida em objetos Java é através de serialização.
- Serializar a informação, é armazená-la junto com informação suficiente para poder reconstruí-la.
- O `stream` de bytes de um objeto salvo, inclui informação suficiente para que os campos, atributos, métodos, assim como todo o objeto possa ser reconstruído.

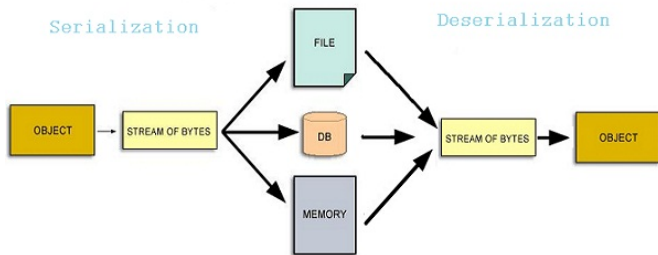


Serialização / DeSerialização

- A chave para armazenar e recuperar a informação contida em objetos Java é através de serialização.
- Serializar a informação, é armazená-la junto com informação suficiente para poder reconstruí-la.
- O `stream` de bytes de um objeto salvo, inclui informação suficiente para que os campos, atributos, métodos, assim como todo o objeto possa ser reconstruído.
- Um `stream` de bytes representa um fluxo de dados, ou um canal de comunicação, com processos lendo em um lado do fluxo e outros escrevendo na outra ponta, conceitualmente.



Serialização e Deserialização



Classe Pessoa.java

```
0 import java.io.Serializable;
1
2 public class Pessoa implements Serializable{
3     private String nome;
4     private String cpf;
5
6     public Pessoa(String nome, String cpf) {
7         this.nome = nome;
8         this.cpf = cpf;
9     }
10    public void imprimir(){
11        System.out.println("Nome: " + nome);
12        System.out.println("CPF: " + cpf);
13    }
14 }
```

- A interface `java.io.Serializable` permite que objetos sejam serializados, ou seja, permite que esses objetos sejam gravados em arquivos, transmitidos pela rede, armazenados em memória, etc.



Classe Principal.java

```
14 public class Principal {
15     /* Criando um vetor de tamanho 2 */
16     private Pessoa[] agenda = new Pessoa[2];
17
18     public void salvarEmDisco(){
19         //código próximas lâminas
20     }
21     public void lerDoDisco(){
22         //código próximas lâminas
23     }
24     public static void main(String[] args) {
25         Principal p = new Principal();
26         p.agenda[0] = new Pessoa("João", "123.734.734-33");
27         p.agenda[1] = new Pessoa("Maria", "354.021.456-12");
28         p.salvarEmDisco();
29         p.lerDoDisco();
30     }
31 }//fim da classe
```



Trecho de código para gravar objeto em arquivo

```
31 public void salvarEmDisco() {
32     File arquivo = new File("meusDados.txt");
33     try {
34         FileOutputStream FOS = new FileOutputStream(arquivo);
35         ObjectOutputStream OOS = new ObjectOutputStream(FOS);
36
37         // gravando o vetor 'pessoa' no arquivo chamado '
38             meusDados.txt'
39         OOS.writeObject(this.pessoa);
40
41         OOS.flush(); // limpando dados em buffer
42         OOS.close(); // fechando fluxo de saída
43         FOS.close(); // fechando arquivo
44     } catch (IOException ex) {
45         System.err.println("erro: " + ex.toString());
46     }
}
```



Trecho de código para ler objeto de um arquivo

```
46 public void lerDoDisco() {
47     File arquivo = new File("meusDados.txt");
48     try {
49         FileInputStream FIS = new FileInputStream(arquivo);
50         ObjectInputStream OIS = new ObjectInputStream(FIS);
51
52         // Lendo os objetos de um arquivo
53         this.pessoa = (Pessoa[]) OIS.readObject();
54         OIS.close(); //fechando fluxo de entrada
55         FIS.close(); //fechando arquivo
56
57         // Uma forma de diferente do for para percorrer vetores
58         for (Pessoa p : this.pessoa) {
59             p.imprimir();
60         }
61     } catch (ClassNotFoundException ex) {
62         System.err.println("erro: " + ex.toString());
63     } catch (IOException ex) {
64         System.err.println("erro: " + ex.toString());
65     }
```

Escrevendo e Lendo em arquivos.

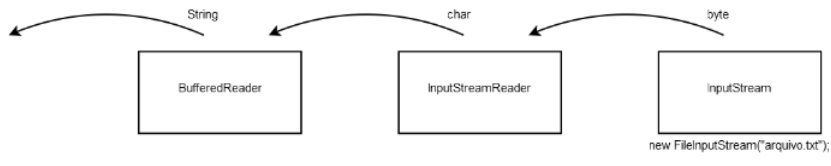
- Crie outra classe Principal, chamada Principal2 e altere os métodos lerDoDisco e salvarEmDisco.
- Copie os trechos de código das próximas lâminas e substitua no arquivo Principal2.
- Rode uma vez, e em seguida comente o método salvarEmDisco, altere o conteúdo do arquivo meusDados.txt manualmente.
- Após alterado, rode o programa.



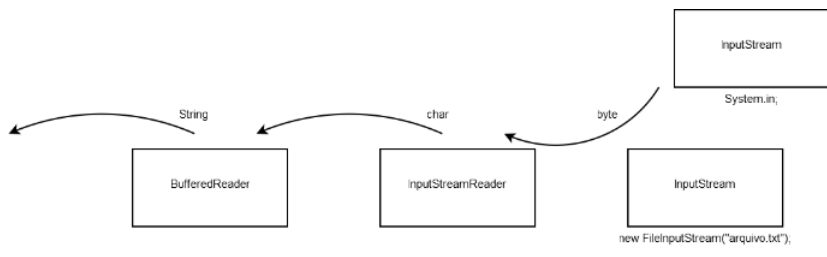
Trecho de código para ler conteúdo de um arquivo

```
65 public void lerDoDisco() {
66     try {
67         InputStream is = new FileInputStream("meusDados.txt");
68         //InputStream is = System.in;
69         InputStreamReader isr = new InputStreamReader(is);
70         BufferedReader br = new BufferedReader(isr);
71
72         String s = br.readLine(); // primeira linha
73         while (s != null) {
74             System.out.println(s);
75             s = br.readLine();
76         }
77         br.close();
78     }catch(java.io.FileNotFoundException FNE){
79         System.out.println("Arquivos não encontrado!");}
80     catch (IOException ex) {
81         System.err.println("erro: " + ex.toString());}
82 }
```

Composição da classe para ler conteúdo de um arquivo



Composição da classe para ler teclado ou conteúdo de um arquivo



Trecho de código para gravar conteúdo em um arquivo

```
82     String toSave = "Informação em texto puro a ser salva.";
83
84     public void salvarEmDisco() {
85         try {
86             OutputStream os = new FileOutputStream("meusDados.txt")
87                 ;
88             OutputStreamWriter osw = new OutputStreamWriter(os);
89             BufferedWriter bw = new BufferedWriter(osw);
90
91             bw.write("Info: "+ this.toSave);
92
93             bw.close();
94         } catch (IOException ex) {
95             System.err.println("erro: " + ex.toString());
96         }
97     }
```



Composição da classe para ler teclado ou conteúdo de um arquivo

