IN1007 Paradigmas de Linguagens de Programação

Itanauã Barbosa (ifb2)
Karla Silva (kmbs)

—

Agenda

- Objetivo
- Contexto
- Exemplos em Java
- BNF
- Exemplo OO1
- Detalhes da implementação na OO1
- Referências

Objetivo

Serialização e deserialização de Objetos para Linguagem Orientada a Objetos 1.

> **write_file**

> **read_file**

Contexto

Serialização de objetos

- Captura o estado do objeto e transformar em uma cadeia de bytes;
- Essa cadeia de bytes inclui: os dados em si e dados sobre o tipo do objeto e seus atributos;
- Existe a “deserialização” para fazer o processo inverso;

Contexto

Como isso é feito em Java?

- A classe precisa implementar a interface **Serializable;**
- Adicionada a partir do JDK 1.1
- Atributos podem ser “não serializáveis” > **transient**

Exemplo em Java

```
import java.io.Serializable;

public class Cliente implements Serializable {
        private String nome;
        private char sexo;
        private String cpf;

        public Cliente(String nome, char sexo, String cpf) {
                super();
                this.nome = nome;
        …

}
```

```
import java.io.FileInputStream;

public class ExemploStream {

    public static void main(String[] args) {

        Cliente cliente = new Cliente("Maria Silva",'F',"0000000001");

        try{

            //Gera o arquivo para armazenar o objeto
            FileOutputStream arquivoGrav = new FileOutputStream("resultado.txt");

            //Classe responsavel por inserir os objetos
            ObjectOutputStream objGravar = new ObjectOutputStream(arquivoGrav);

            //Grava o objeto cliente no arquivo
            objGravar.writeObject(cliente);

            objGravar.flush();
            objGravar.close();
            arquivoGrav.flush();
            arquivoGrav.close();
            System.out.println("Objeto gravado com sucesso! \n");

        }
```

Exemplo em Java

[illegible]srCliente[illegible]q[illegible]~CsexoLcpftLjava/lang/String;Lnomeq~xpFt
0000000001t

Maria Silva

Resultado.txt

Backus–Naur form

Comando ::= Atribuicao
| ComDeclaracao
| While
| IfThenElse
| **IO**
...

IO ::= "write" "(" Expressao ")" | "read" "(" Id ")" | **"write_file" "(" Id "," Expressao ")" | "read_file" "(" Id "," Expressao "," Expressao ")"**

Exemplo

```
{
    classe Cliente {
        string name = "",
        string cpf = "",
        string sexo = ""
        ;

        proc print() {
          write(this.name);
          write(this.sexo);
          write(this.cpf)
        },

        proc setName(string receivedName) {
          this.name := receivedName
        },

        proc setCpf(string receivedCpf) {
          this.cpf := receivedCpf
        },

        proc setSexo(string receivedSexo) {
          this.sexo := receivedSexo
        }
    }

    ;

}
```

Exemplo

```
{
    Cliente c1 := new Cliente,
    Cliente c2 := new Cliente,
    Cliente c3 := new Cliente,
    Cliente c4 := new Cliente
    ;

    write("Teste do write_file + read_file");

    c1.setName("Jose Silva");
    c1.setSexo("M");
    c1.setCpf("111.111.111-01");

    c2.setName("Maria Santos");
    c2.setSexo("F");
    c2.setCpf("222.222.222-02");
```

Exemplo

```
write_file(object, filename);
    write_file(c1, "output1.txt");
    write_file(c2, "output1.txt");

read_file(object to store, filename, index);
    read_file(c3, "output1.txt", 0);
    read_file(c4, "output1.txt", 1);
```

Detalhes da implementação

- As seguintes classes agora implementam a interface **Serializable:** ValorNull, ValorString, … , Objeto, ContextoObject;
- Adicionamos dois novos tokens: **write_file** e **read_file;**
- Adicionamos o **PWriteFile** e o **PReadFile;**
- Adicionamos as duas “funções” acima ao **PIO;**

Detalhes da implementação

- Write File:
  - Recebe um atributo **Id** e um **Expressao**: objeto recebido, e o nome do arquivo;
  - Se o arquivo já existe, é feito um append; Caso não, um novo arquivo é criado;
  - Foi adicionada uma nova classe chamada **AppendingObjectOutputStream** para fazer o append;
  - Obtemos o objeto do tipo **Objeto** do ambiente a partir da Id (objeto) recebido;
  - Fechamos o ObjectOutputStream em cada chamada do write_file;
  - Retornamos o ambiente sem modificações;

Detalhes da implementação

Read File:

- Recebe um atributo **Id** e dois atributos **Expressao**: objeto que deverá receber o objeto lido do arquivo, o nome do arquivo, e a posição dele na lista;
- Recuperamos todos os objetos do arquivo como "**Objeto**" e usamos apenas o do índice desejado;
- Obtemos um proxRef;
- Mapeamos o objeto do índice desejado no HashMap com a proxRef;
- Fazemos a troca de valor entre o Id do objeto que vai receber o arquivo e a proxRef;
- Retornamos o ambiente sem modificações;
- Fechamos o FileInputStream e o ObjectInputStream em cada chamada do ReadFile;
- Retornamos o ambiente sem modificações;

Referências

- Linguagem orientada a objetos 1: https://www.cin.ufpe.br/~in1007/linguagens/OrientadaObjetos1/orientadaObjetos1.html
- Interface Serializable: https://docs.oracle.com/javase/7/docs/api/java/io/Serializable.html
- Repositório no GitHub: https://github.com/karlambsilva/plp-project