08

Mãos na massa: Serialização e deserialização de objetos

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

Serialização

1) No projeto **java-io**, crie a classe **TesteSerializacao**, no pacote **br.com.alura.java.io.teste**, com o método main:

```
public class TesteSerializacao {

    public static void main(String[] args) {

    }

}
```

2) Crie uma string e instancie um `ObjectOutputStream` passando como parâmetro para o seu construtor um `FileOutputStream` de nome **objeto.bin**. Trate a exceção através de um `throws` :

```
public class TesteSerializacao {

    public static void main(String[] args) throws IOException {

        String nome = "Nico Steppat";

        ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("objeto.bin"));

    }

}
```

3) Escreva a string no objeto e feche-o:

```
public class TesteSerializacao {

    public static void main(String[] args) throws IOException {

        String nome = "Nico Steppat";

        ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("objeto.bin"));
        oos.writeObject(nome);
        oos.close();

    }

}
```

Execute e atualize o projeto, veja que o **objeto.bin** foi criado.

4) Comente ou apague todo o código anterior. Instancie um `ObjectInputStream` passando como parâmetro para o seu construtor um `FileInputStream`, o **objeto.bin**:

```
public class TesteSerializacao {

    public static void main(String[] args) throws IOException {

        ObjectInputStream ois = new ObjectInputStream(new FileInputStream("objeto.bin"));

    }

}
```

5) Leia o objeto e guarde-o em uma string, fazendo um *cast*. Trate a exceção através de um `throws` e em seguida, feche o `ObjectInputStream` e imprima a string:

```
public class TesteSerializacao {

    public static void main(String[] args) throws IOException, ClassNotFoundException {

        ObjectInputStream ois = new ObjectInputStream(new FileInputStream("objeto.bin"));
        String nome = (String) ois.readObject();
        ois.close();
        System.out.println(nome);

    }

}
```

Serialização de qualquer objeto

6) Caso você não tenha, baixe a classe `Cliente` aqui (https://caelum-online-public.s3.amazonaws.com/857-java-io/06/Cliente.java) e copie-a para dentro do projeto **java-io**, no pacote **br.com.alura.java.io.teste**.

7) No projeto **java-io**, no pacote **br.com.alura.java.io.teste**, copie a classe `TesteSerializacao`, dando o nome `TesteSerializacaoCliente`. Após isso, na classe `TesteSerializacaoCliente`, instancie e popule um `Cliente` e comente ou apague as linhas de criação do `ObjectInputStream` em diante:

```
public class TesteSerializacaoCliente {

    public static void main(String[] args) throws IOException, ClassNotFoundException {

        Cliente cliente = new Cliente();
        cliente.setNome("Nico");
        cliente.setProfissao("Dev");
        cliente.setCpf("234113131");

    }

}
```

8) Adicione as linhas de escrita implementadas anteriormente. Faça as devidas alterações e dessa vez escreva no arquivo **cliente.bin**:

```
public class TesteSerializacaoCliente {

    public static void main(String[] args) throws IOException, ClassNotFoundException {

        Cliente cliente = new Cliente();
        cliente.setNome("Nico");
        cliente.setProfissao("Dev");
        cliente.setCpf("234113131");

        ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("cliente.bin"));
        oos.writeObject(cliente);
        oos.close();

    }

}
```

Ao executar a classe, perceba que uma exceção ocorre.

9) Para funcionar, faça com que a classe `Cliente` *implemente* a classe `Serializable` e adicione o `serialVersionUID` padrão:

```
public class Cliente implements Serializable {

    private static final long serialVersionUID = 1L;

    // restante do código omitido

}
```

Ao executar novamente a classe `TesteSerializacaoCliente` , o arquivo **cliente.bin** é criado.

10) Ainda na classe `TesteSerializacaoCliente` , apague ou comente todo o código e adicione as linhas de leitura implementadas anteriormente. Faça as devidas alterações, e dessa vez leia o arquivo **cliente.bin** e imprima o CPF do cliente:

```
public class TesteSerializacaoCliente {

    public static void main(String[] args) throws IOException, ClassNotFoundException {

        ObjectInputStream ois = new ObjectInputStream(new FileInputStream("cliente.bin"));
        Cliente cliente = (Cliente) ois.readObject();
        ois.close();
        System.out.println(cliente.getCpf());

    }

}
```

Execute e veja o CPF do cliente sendo impresso.

Serialização com herança

11) Se você não o tem, baixe o projeto **bytebank-herdado-conta** aqui (https://caelum-online-public.s3.amazonaws.com/857-java-io/06/bytebank-herdado-conta.zip).

12) No projeto **bytebank-herdado-conta**, crie a classe `TesteSerializacao`, no pacote **br.com.bytebank.banco.test.io**, com o método `main`:

```
public class TesteSerializacao {

    public static void main(String[] args) {

    }

}
```

13) Adicione o código de escrita feito anteriormente, crie um cliente e uma conta corrente, modifique o arquivo de escrita e salve a conta:

```
public class TesteSerializacao {

    public static void main(String[] args) throws FileNotFoundException, IOException {

        Cliente cliente = new Cliente();
        cliente.setNome("Nico");
        cliente.setProfissao("Dev");
        cliente.setCpf("234113131");

        ContaCorrente cc = new ContaCorrente(222, 333);
        cc.setTitular(cliente);
        cc.deposita(222.3);

        ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("cc.bin"));
        oos.writeObject(cc);
        oos.close();

    }


}
```

14) Ao executar a classe, um erro ocorre, pois a classe `ContaCorrente` utiliza o construtor da classe-mãe, `Conta`, que não está implementando `Serializable`. Então, faça isso:

```
public abstract class Conta extends Object implements Comparable<Conta>, Serializable {

    // restante do código omitido
}
```

15) Faça o mesmo para a classe `Cliente`:

```
public class Cliente implements Serializable {

    // restante do código omitido
}
```

Ao executar novamente a classe `TesteSerializacao`, o arquivo **cc.bin** é gerado.

16) Ainda no projeto **bytebank-herdado-conta**, crie a classe `TesteDeserializacao`, no pacote **br.com.bytebank.banco.test.io**, com o método `main`:

```
public class TesteDeserializacao {

    public static void main(String[] args) {

    }

}
```

17) Adicione o código de leitura feito anteriormente, modificando o arquivo a ser lido. Leia a conta, guarde-a em uma variável e imprima seu saldo e o nome do titular:

```
public class TesteDeserializacao {

    public static void main(String[] args) throws FileNotFoundException, IOException, ClassNotFoundE

        ObjectInputStream ois = new ObjectInputStream(new FileInputStream("cc.bin"));
        ContaCorrente cc = (ContaCorrente) ois.readObject();
        ois.close();

        System.out.println(cc.getSaldo());
        System.out.println(cc.getTitular().getNome());

    }

}
```

Ao executar a classe, os dados arquivo **cc.bin** são lidos corretamente.