Orientação a objetos

Serialização

Serialização

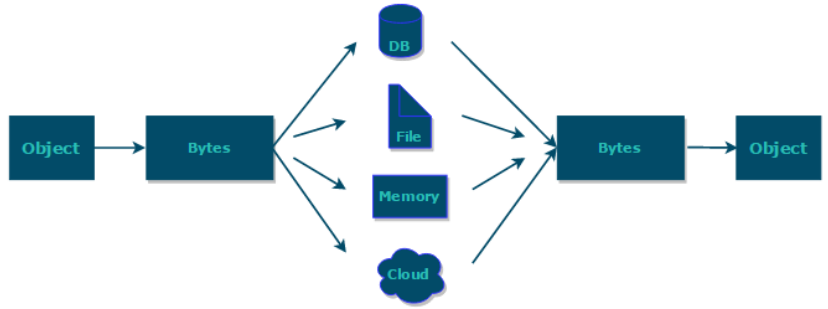
- A serialização é o processo de converter um objeto em um fluxo de bytes para armazenar o objeto ou transmiti-lo para a memória, um banco de dados ou um arquivo.

- Sua finalidade principal é salvar o estado de um objeto para recriá-lo quando necessário.

- O processo inverso é chamado desserialização.

Serialização

- O objeto é serializado para um fluxo que carrega os dados.
- O fluxo também pode ter informações sobre o tipo do objeto, como sua versão, cultura e nome do assembly.
- Desse fluxo, o objeto pode ser armazenado em um banco de dados, um arquivo ou memória.



Serialização

- O processo de serialização é independente da aplicação, um dado serializado em uma plataforma deve poder ser deserializada por qualquer outra.

- Visando garantir a comunicação entre aplicações.

Serialização

- Inicialmente, quando a capacidade de armazenamento das máquinas e transferência de dados na rede era baixo, a serialização era feita convertendo os objetos/estruturas de dados para *stream* de *bytes*, *byte-stream-based encoding*.

Serialização

- Com o avanço tecnológico, permitiu-se o uso de um padrão não tão compacto, quanto um simples *streams* de bytes, surgindo a proposta de se padronizar o uso para XML, um *text-based encoding*, que tem a vantagem de ser compreensível a humanos, pode ser aberto em leitor de texto simples e cumprir o propósito de ser entendível pelo programas, independente de sua linguagem.

- Depois sugiram alternativas mais leves e de leitura mais amigáveis para humanos como JSON, que hoje é o mais utilizado, e o YAML.

Exemplo

```
5  class Produto implements Serializable {
6      private String codigo;
7      private String nome;
8      private double preco;
9      private transient String temporario;
10
11     public Produto(String codigo, String nome, double preco, String temporario) {
12         this.codigo = codigo;
13         this.nome = nome;
14         this.preco = preco;
15         this.temporario = temporario;
16     }
17
18     public String getCodigo() {
19         return codigo;
20     }
21
22     public String getNome() {
23         return nome;
24     }
25
26     public double getPreco() {
27         return preco;
28     }
29
30     public String getTemporario() {
31         return temporario;
32     }
33
34     @Override
35     public String toString() {
36         return "Produto [codigo=" + codigo + ", nome=" + nome + ", preco=" + preco + "]";
37     }
38 }
```

```java
class Produto implements Serializable {
    private String codigo;
    private String nome;
    private double preco;
    private transient String temporario;

    public Produto(String codigo, String nome, double preco, String temporario) {
        this.codigo = codigo;
        this.nome = nome;
        this.preco = preco;
        this.temporario = temporario;
    }

    public String getCodigo() {
        return codigo;
    }

    public String getNome() {
        return nome;
    }

    public double getPreco() {
        return preco;
    }

    public String getTemporario() {
        return temporario;
    }

    @Override
    public String toString() {
        return "Produto [codigo=" + codigo + ", nome=" + nome + ", preco=" + preco + "]";
    }
}
```

Exemplo

```
9  public class Principal {
10     public static void main(String[] args) {
11         Produto produto = new Produto("ABC123", "Exemplo de Produto", 9.99, "Campo temporário");
12
13         // Serialização
14         try {
15             FileOutputStream arquivoSaida = new FileOutputStream("produto.ser");
16             ObjectOutputStream objetoSaida = new ObjectOutputStream(arquivoSaida);
17
18             objetoSaida.writeObject(produto);
19             objetoSaida.close();
20             arquivoSaida.close();
21
22             System.out.println("Objeto serializado e salvo em produto.ser");
23         } catch (IOException e) {
24             e.printStackTrace();
25         }
26
27         // Desserialização
28         try {
29             FileInputStream arquivoEntrada = new FileInputStream("produto.ser");
30             ObjectInputStream objetoEntrada = new ObjectInputStream(arquivoEntrada);
31
32             Produto produtoDesserializado = (Produto) objetoEntrada.readObject();
33             objetoEntrada.close();
34             arquivoEntrada.close();
35
36             System.out.println("Objeto desserializado: " + produtoDesserializado); //chama o método sobrescrito toString();
37             System.out.println("Vai apresentar NULL: "+produtoDesserializado.getTemporario());
38         } catch (IOException | ClassNotFoundException e) {
39             e.printStackTrace();
40         }
41     }
42 }
```

```java
public class Principal {
    public static void main(String[] args) {
        Produto produto = new Produto("ABC123", "Exemplo de Produto", 9.99, "Campo temporário");

        // Serialização
        try {
            FileOutputStream arquivoSaida = new FileOutputStream("produto.ser");
            ObjectOutputStream objetoSaida = new ObjectOutputStream(arquivoSaida);

            objetoSaida.writeObject(produto);
            objetoSaida.close();
            arquivoSaida.close();

            System.out.println("Objeto serializado e salvo em produto.ser");
        } catch (IOException e) {
            e.printStackTrace();
        }

        // Desserialização
        try {
            FileInputStream arquivoEntrada = new FileInputStream("produto.ser");
            ObjectInputStream objetoEntrada = new ObjectInputStream(arquivoEntrada);

            Produto produtoDesserializado = (Produto) objetoEntrada.readObject();
            objetoEntrada.close();
            arquivoEntrada.close();

            System.out.println("Objeto desserializado: " + produtoDesserializado); //chama o método sobrescrito toString();
            System.out.println("Vai apresentar NULL: "+produtoDesserializado.getTemporario());
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

Serialização Json

```
2  public class Pessoa {
3      private String nome;
4      private int idade;
5
6      public Pessoa(String nome, int idade) {
7          super();
8          this.nome = nome;
9          this.idade = idade;
10     }
11
12     public String getNome() {
13         return nome;
14     }
15
16     public int getIdade() {
17         return idade;
18     }
19
20     @Override
21     public String toString() {
22         return "Pessoa{nome=" + nome + ", idade=" + idade + "}";
23     }
24 }
```

```java
public class Pessoa {
    private String nome;
    private int idade;

    public Pessoa(String nome, int idade) {
        super();
        this.nome = nome;
        this.idade = idade;
    }

    public String getNome() {
        return nome;
    }

    public int getIdade() {
        return idade;
    }

    @Override
    public String toString() {
        return "Pessoa{nome=" + nome + ", idade=" + idade + "}";
    }
}
```

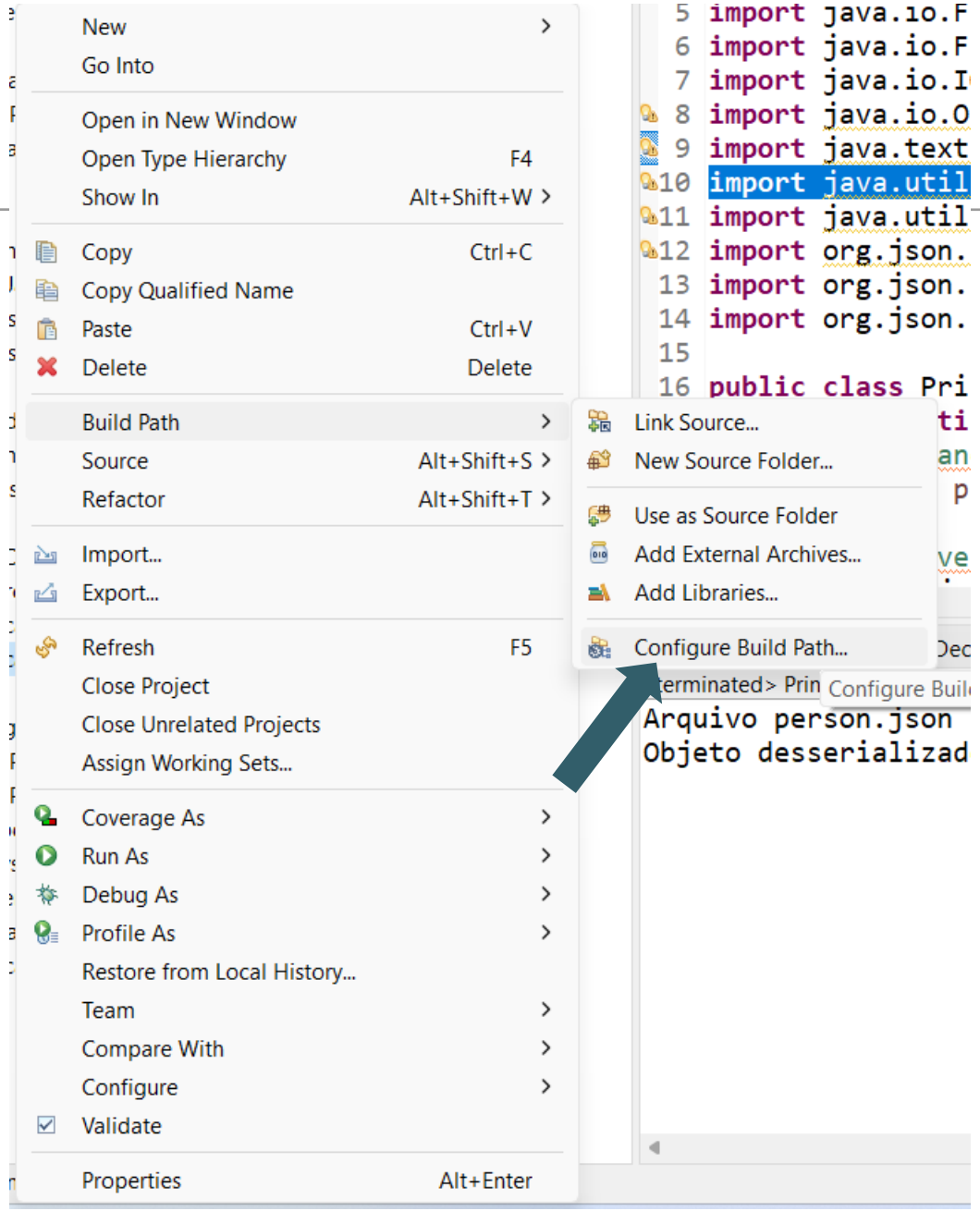
Serialização Json

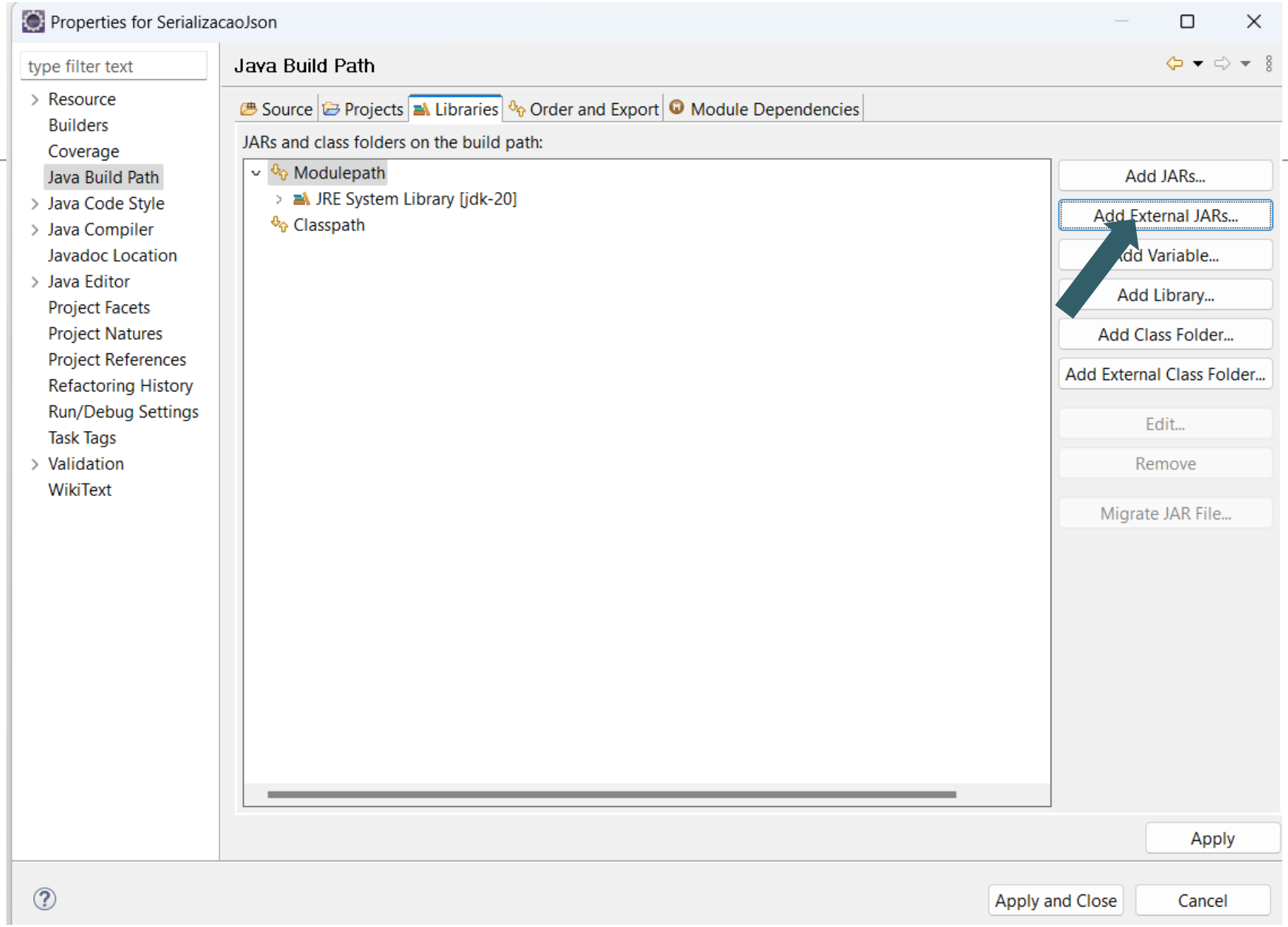
```
16 public class Principal {
17     public static void main(String[] args) {
18         // Criando um objeto para serializar
19         Pessoa p = new Pessoa("Ricardo", 30);
20 
21         // Convertendo o objeto em um JSONObject
22         JSONObject json = new JSONObject();
23         json.put("nome", p.getNome());
24         json.put("idade", p.getIdade());
25         String jsonString = json.toJSONString();
26 
27         gravaArquivo(jsonString);
28         try {
29             lerArquivo();
30         } catch (org.json.simple.parser.ParseException e) {
31             // TODO Auto-generated catch block
32             e.printStackTrace();
33         }
34     }
```

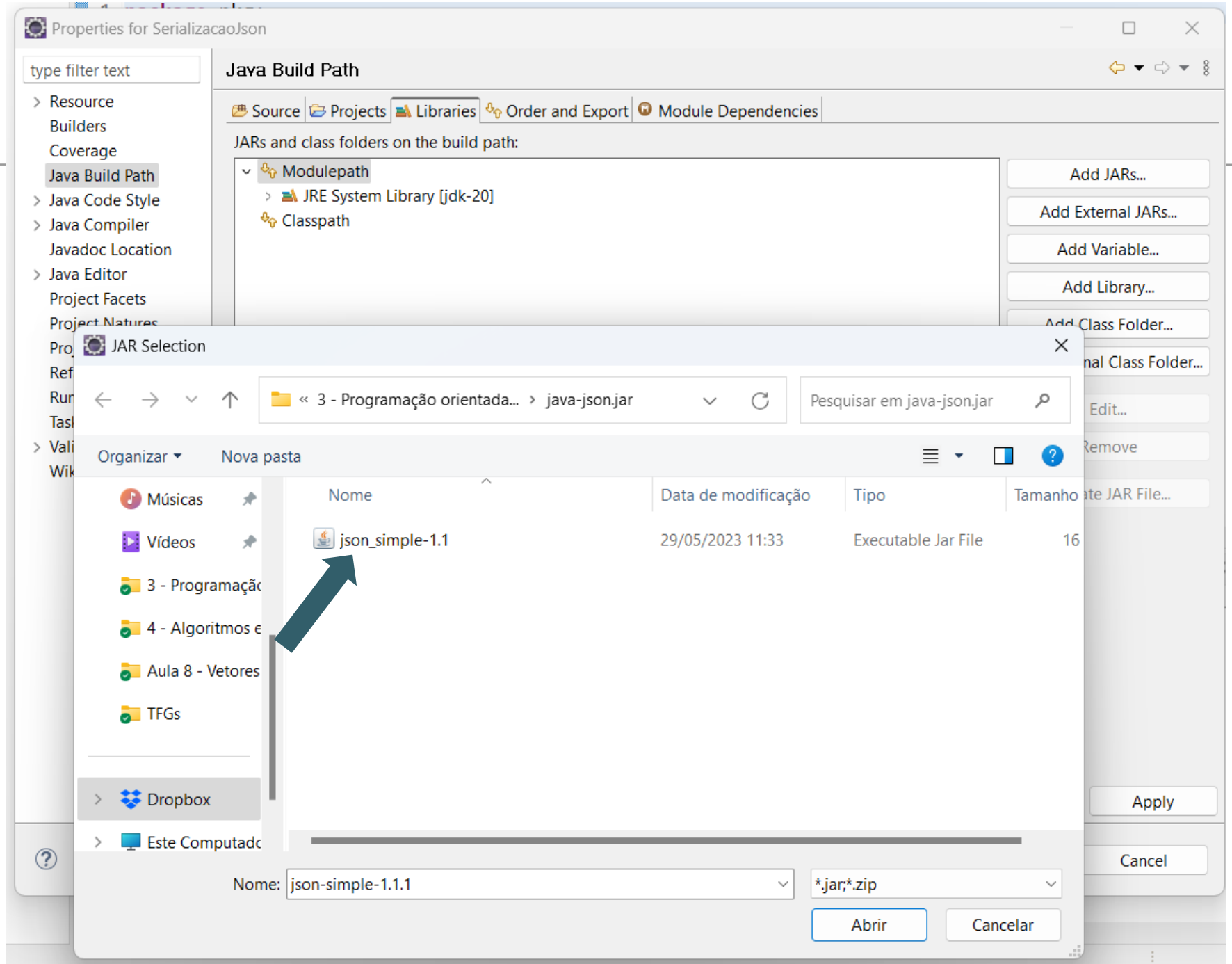
```java
public class Principal {
    public static void main(String[] args) {
        // Criando um objeto para serializar
        Pessoa p = new Pessoa("Ricardo", 30);

        // Convertendo o objeto em um JSONObject
        JSONObject json = new JSONObject();
        json.put("nome", p.getNome());
        json.put("idade", p.getIdade());
        String jsonString = json.toJSONString();

        gravaArquivo(jsonString);
        try {
            lerArquivo();
        } catch (org.json.simple.parser.ParseException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```


New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
Close Project
Close Unrelated Projects
Assign Working Sets...
Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Team
Compare With
Configure
Validate
Properties Alt+Enter
Link Source...
New Source Folder...
Use as Source Folder
Add External Archives...
Add Libraries...
Configure Build Path...
Configure Buil
import java.io.F
import java.io.F
import java.io.I
import java.io.O
import java.text
import java.util
import java.util
import org.json.
import org.json.
import org.json.
public class Pri
terminated> Prin
Arquivo person.json
Objeto desserializad


Properties for SerializacaoJson
type filter text
Resource
Builders
Coverage
Java Build Path
Java Code Style
Java Compiler
Javadoc Location
Java Editor
Project Facets
Project Natures
Project References
Refactoring History
Run/Debug Settings
Task Tags
Validation
WikiText
Java Build Path
Source
Projects
Libraries
Order and Export
Module Dependencies
JARs and class folders on the build path:
Modulepath
JRE System Library [jdk-20]
Classpath
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Add External Class Folder...
Edit...
Remove
Migrate JAR File...
Apply
Apply and Close
Cancel


Properties for SerializacaoJson
type filter text
Resource
Builders
Coverage
Java Build Path
Java Code Style
Java Compiler
Javadoc Location
Java Editor
Project Facets
Java Build Path
Source
Projects
Libraries
Order and Export
Module Dependencies
JARs and class folders on the build path:
Modulepath
JRE System Library [jdk-20]
Classpath
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Edit...
Remove
Apply
Cancel
JAR Selection
3 - Programação orientada...
java-json.jar
Pesquisar em java-json.jar
Organizar
Nova pasta
Músicas
Vídeos
3 - Programaçã
4 - Algoritmos e
Aula 8 - Vetores
TFGs
Dropbox
Este Computado
Nome
Data de modificação
Tipo
Tamanho
json_simple-1.1
29/05/2023 11:33
Executable Jar File
16
Nome:
json-simple-1.1.1
.jar;.zip
Abrir
Cancelar

Serialização Json

```
12 import org.json.simple.JSONArray;
13 import org.json.simple.JSONObject;
14 import org.json.simple.parser.JSONParser;
```

Serialização Json

```
public static void gravaArquivo(String jsonString) {

     try (FileWriter fileWriter = new FileWriter("pessoa.json")) {
         fileWriter.write(jsonString);
         System.out.println("Arquivo person.json salvo com sucesso.");
     } catch (IOException e) {
         e.printStackTrace();
     }
}

public static void lerArquivo() throws org.json.simple.parser.ParseException {
// Lendo o arquivo e desserializando o JSON para objeto
    try (FileReader fileReader = new FileReader("pessoa.json")) {
        JSONParser jsonParser = new JSONParser();
        JSONObject jsonObject = (JSONObject) jsonParser.parse(fileReader);

        // Criando um objeto Person a partir do JSON
        String nome = (String) jsonObject.get("nome");
        long idade = (long) jsonObject.get("idade");
        Pessoa deserializedPerson = new Pessoa(nome, (int) idade);

        System.out.println("Objeto desserializado: " + deserializedPerson);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

```java
public static void gravaArquivo(String jsonString) {

     try (FileWriter fileWriter = new FileWriter("pessoa.json")) {
         fileWriter.write(jsonString);
         System.out.println("Arquivo person.json salvo com sucesso.");
     } catch (IOException e) {
         e.printStackTrace();
     }
}

public static void lerArquivo() throws org.json.simple.parser.ParseException {
// Lendo o arquivo e desserializando o JSON para objeto
    try (FileReader fileReader = new FileReader("pessoa.json")) {
        JSONParser jsonParser = new JSONParser();
        JSONObject jsonObject = (JSONObject) jsonParser.parse(fileReader);

        // Criando um objeto Person a partir do JSON
        String nome = (String) jsonObject.get("nome");
        long idade = (long) jsonObject.get("idade");
        Pessoa deserializedPerson = new Pessoa(nome, (int) idade);

        System.out.println("Objeto desserializado: " + deserializedPerson);
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```