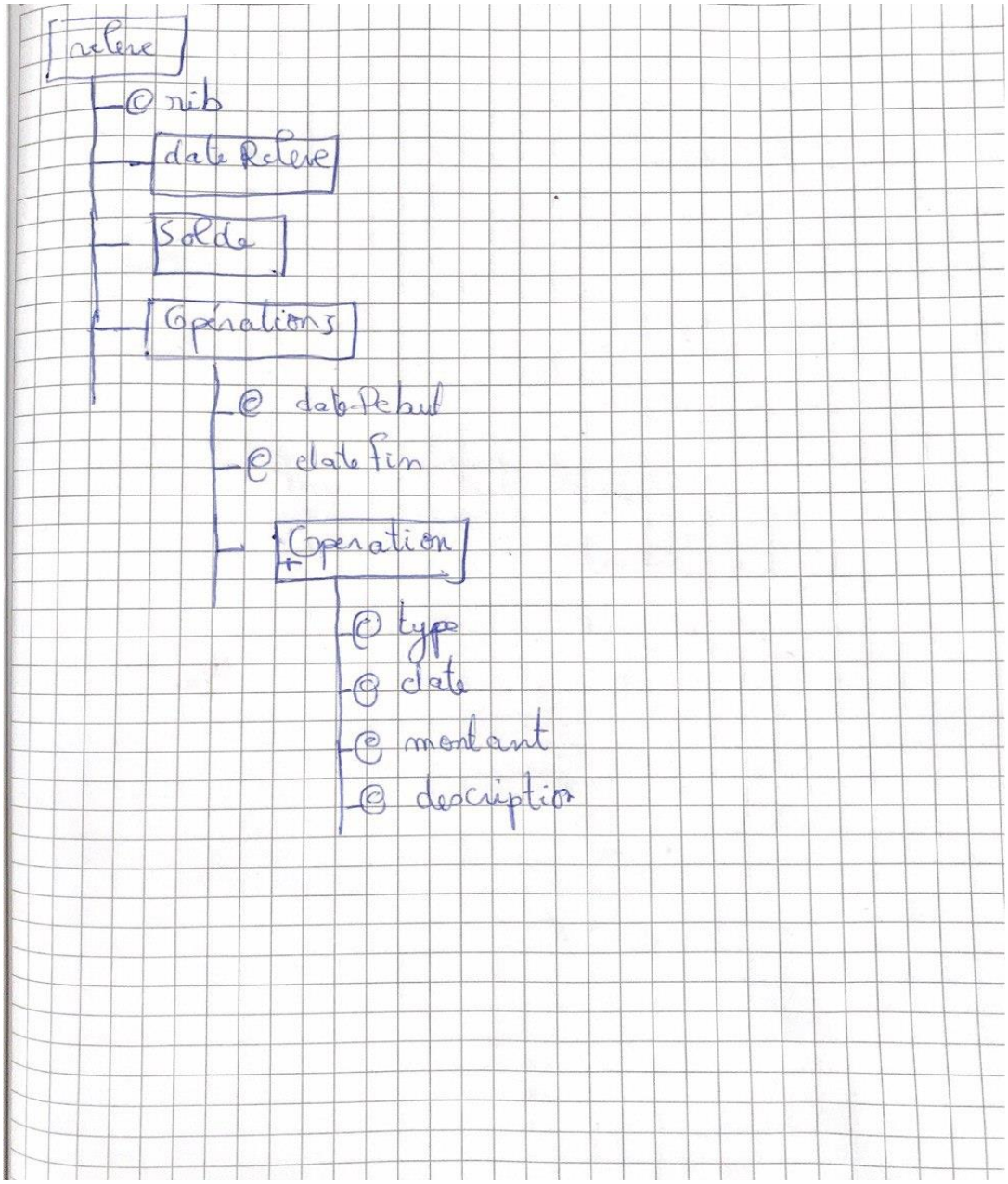
Rapport Projet Technologie XML et Web Services

Introduction

Dans ce projet, on souhaite créer une application qui permet de gérer les relevés de comptes bancaires.

A) Partie Technologie XML

1) Structure graphique de l'arbre XML



2) DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT releve (dateReleve, solde, operations) >
<!ELEMENT dateReleve (#PCDATA) >
<!ELEMENT solde (#PCDATA) >
<!ELEMENT operations (operation+) >
<!ELEMENT operation (#PCDATA) >

<!ATTLIST releve
              rib NMTOKEN #REQUIRED>

<!ATTLIST operations
              dateDebut CDATA #REQUIRED
              dateFin CDATA #REQUIRED>

<!ATTLIST operation
              type CDATA #REQUIRED
              date CDATA #REQUIRED
              montant NMTOKEN #REQUIRED
              description CDATA #REQUIRED>
```

- **Document XML validé par la DTD**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE releve SYSTEM "file:/C:/Users/Baba%20Prince/Desktop/Examen%20Technologie%20XML%20-%20Web%20Services/Te
   <releve rib="01111222233334444555566">
    <dateReleve>2021-11-10</dateReleve>
    <solde>14500</solde>
    <operations dateDebut="2021-01-01" dateFin="2021-01-30">
        <operation type="CREDIT" date="2021-01-01" montant="9000" description="Vers Espèce"></operation>
    </operations>
</releve>
```

3) XSD

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
<xsd:element name="releve">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="dateReleve" type="xsd:date"></xsd:element>
            <xsd:element name="solde" type="xsd:double"></xsd:element>
            <xsd:element name="operations" type="OPS"></xsd:element>
        </xsd:sequence>
        <xsd:attribute name="rib" type="xsd:string" use="required"></xsd:attribute>
    </xsd:complexType>
</xsd:element>

<xsd:complexType name="OPS">
    <xsd:sequence>
        <xsd:element name="operation" type="OP" maxOccurs="unbounded"></xsd:element>
    </xsd:sequence>
    <xsd:attribute name="dateDebut" type="xsd:date" use="required"></xsd:attribute>
    <xsd:attribute name="dateFin" type="xsd:date" use="required"></xsd:attribute>
</xsd:complexType>

<xsd:complexType name="OP">
    <xsd:attribute name="type" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="credit"></xsd:enumeration>
                <xsd:enumeration value="debit"></xsd:enumeration>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="date" type="xsd:date" use="required"></xsd:attribute>
    <xsd:attribute name="montant" type="xsd:double" use="required"></xsd:attribute>
    <xsd:attribute name="description" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="vers espèce"></xsd:enumeration>
                <xsd:enumeration value="chèque guichet"></xsd:enumeration>
                <xsd:enumeration value="prélèvement assurance"></xsd:enumeration>
                <xsd:enumeration value="virement"></xsd:enumeration>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>
</xsd:schema>

```

- **Document XML validé par la XSD**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<releve xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="file:/C:/Users/Baba%20Prince/Desktop/Examen%20Technologie%20XML%20-%20Web%20Services/Technologie%2
    rib="01111222233334444555666">
    <dateReleve>2021-11-10</dateReleve>
    <solde>14500</solde>
    <operations dateDebut="2021-01-01" dateFin="2021-01-30">
        <operation type="credit" date="2021-01-01" montant="9000" description="vers espèce"/>
        <operation type="credit" date="2021-02-02" montant="5000" description="vers espèce"/>
        <operation type="debit" date="2021-03-02" montant="7000" description="chèque guichet"/>
    </operations>
</releve>
```

4) XSL 1

Cette feuille de style permet d'afficher les toutes les données de ce document

XML au format HTML en affichant le total des opérations de débit et le total des opérations de crédit.

```xml
<xsl:template match="/">
    <html>
        <head>
            <title>Relevé</title>
        </head>
        <body>
            <h1>Releves</h1>
            <xsl:for-each select="releve">
                <p>Date relevé : <xsl:value-of select="dateReleve"></xsl:value-of> </p>
                <p>RIB : <xsl:value-of select="@rib"></xsl:value-of> </p>
                <p>Solde <xsl:value-of select="solde"></xsl:value-of> </p>
                <table border="2">
                    <caption>Somme des Opérations</caption>
                    <thead>
                        <th>Type d'opération</th>
                        <th>Date</th>
                        <th>Montant</th>
                        <th>Description</th>
                    </thead>
                    <tbody>
                        <xsl:for-each select="operations">
                            <xsl:for-each select="operation">
                                <tr>
```

```xml
                    <tbody>
                        <xsl:for-each select="operations">
                            <xsl:for-each select="operation">
                                <tr>
                                    <td> <xsl:value-of select="@type"></xsl:value-of> </td>
                                    <td> <xsl:value-of select="@date"></xsl:value-of> </td>
                                    <td> <xsl:value-of select="@montant"></xsl:value-of> </td>
                                    <td> <xsl:value-of select="@description"></xsl:value-of> </td>
                                </tr>
                            </xsl:for-each>
                            <tr>
                                <td colspan="2">Total des operations de crédit</td>
                                <td colspan="2" align="center"> <xsl:value-of select="sum(operation[@type='credit']/@montant)"></xsl:value-o
                            </tr>
                            <tr>
                                <td colspan="2">Total des operations de débit</td>
                                <td colspan="2" align="center"> <xsl:value-of select="sum(operation[@type='debit']/@montant)"></xsl:value-of
                            </tr>
                        </xsl:for-each>
                    </tbody>
                </table>
            </xsl:for-each>
        </body>
```

- **Résultat de la feuille de style 1**

Releves

Date relevé : 2021-11-10

RIB : 01111222333344445555666

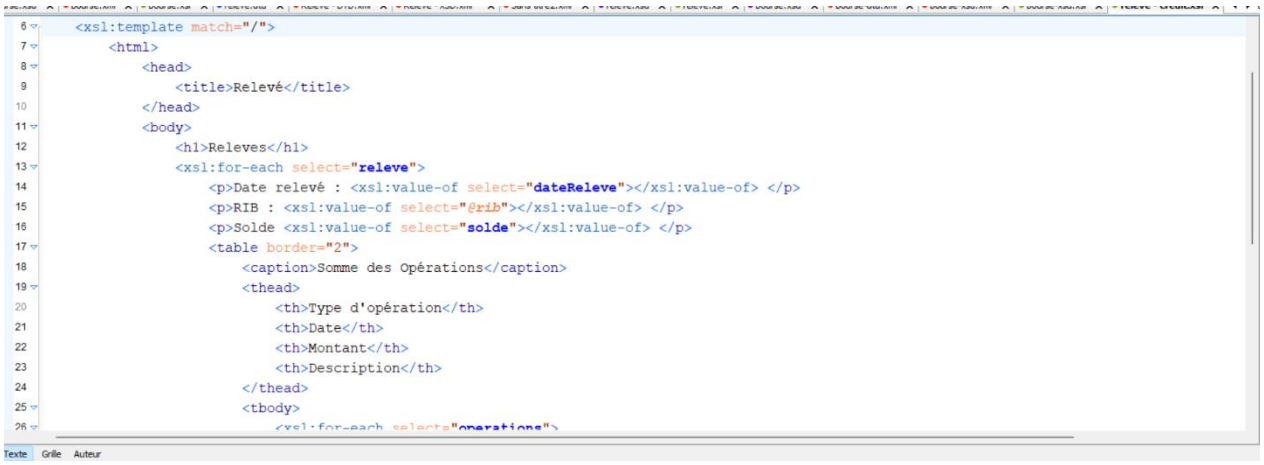
Solde 14500

Somme des Opérations

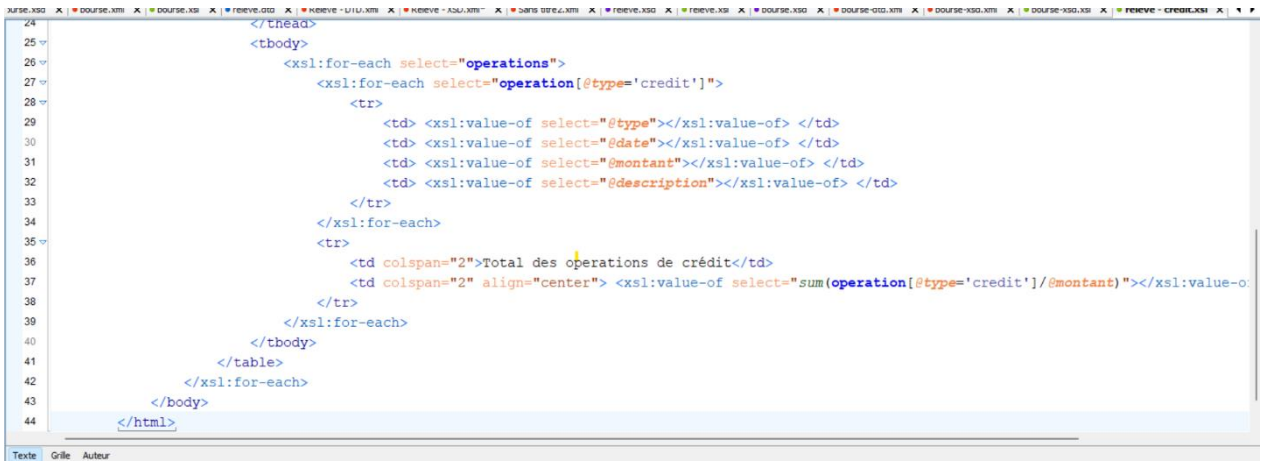
Type d'opération	Date	Montant	Description
credit	2021-01-01	9000	vers espèce
credit	2021-02-02	5000	vers espèce
debit	2021-03-02	7000	chèque guichet
Total des operations de crédit		14000	
Total des operations de débit		7000	

5) XSL 2

Cette feuille de style permet d'afficher au format HTML les opérations de type CREDIT d'un relevé bancaire.



```
<xsl:template match="/">
    <html>
        <head>
            <title>Relevé</title>
        </head>
        <body>
            <h1>Releves</h1>
            <xsl:for-each select="releve">
                <p>Date relevé : <xsl:value-of select="dateReleve"></xsl:value-of> </p>
                <p>RIB : <xsl:value-of select="@rib"></xsl:value-of> </p>
                <p>Solde <xsl:value-of select="solde"></xsl:value-of> </p>
                <table border="2">
                    <caption>Somme des Opérations</caption>
                    <thead>
                        <th>Type d'opération</th>
                        <th>Date</th>
                        <th>Montant</th>
                        <th>Description</th>
                    </thead>
                    <tbody>
                        <xsl:for-each select="operations">
```



```
                    </thead>
                    <tbody>
                        <xsl:for-each select="operations">
                            <xsl:for-each select="operation[@type='credit']">
                                <tr>
                                    <td> <xsl:value-of select="@type"></xsl:value-of> </td>
                                    <td> <xsl:value-of select="@date"></xsl:value-of> </td>
                                    <td> <xsl:value-of select="@montant"></xsl:value-of> </td>
                                    <td> <xsl:value-of select="@description"></xsl:value-of> </td>
                                </tr>
                            </xsl:for-each>
                            <tr>
                                <td colspan="2">Total des operations de crédit</td>
                                <td colspan="2" align="center"> <xsl:value-of select="sum(operation[@type='credit']/@montant)"></xsl:value-o
                            </tr>
                        </xsl:for-each>
                    </tbody>
                </table>
            </xsl:for-each>
        </body>
    </html>
```

- **Resultat de la feuille de style 2**

Releves

Date relevé : 2021-11-10

RIB : 01111222233344445555666

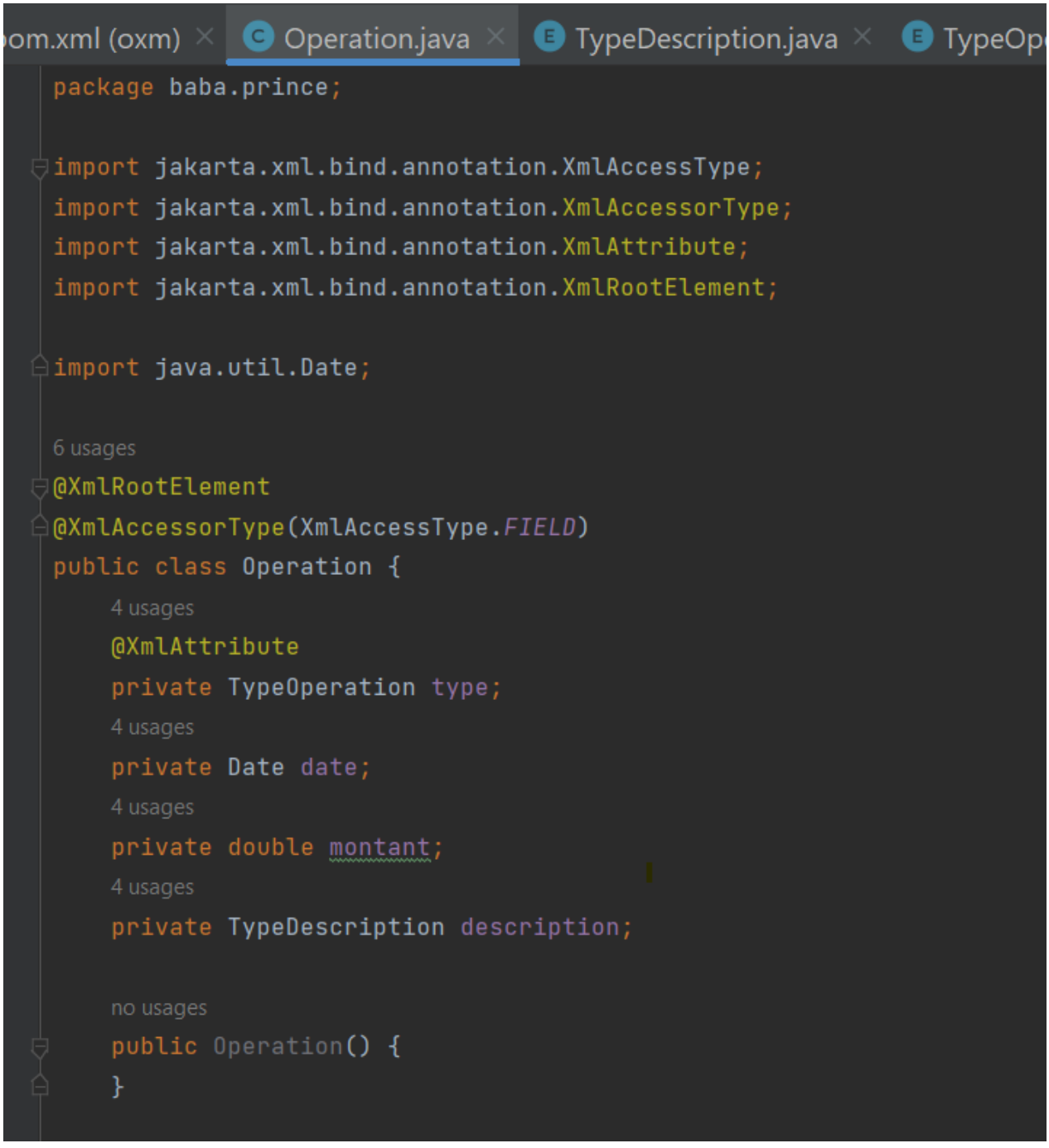
Solde 14500

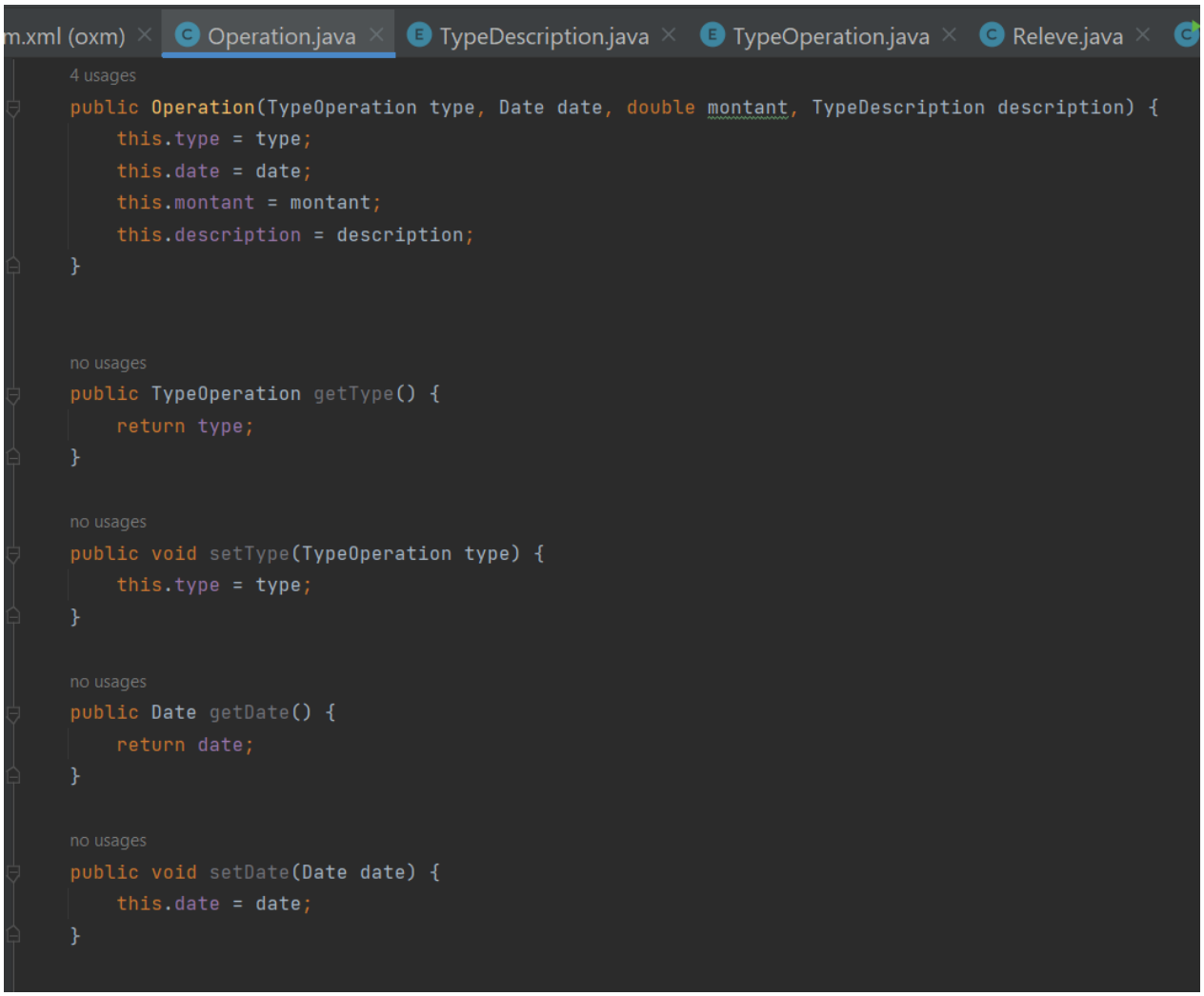
Somme des Opérations

Type d'opération	Date	Montant	Description
credit	2021-01-01	9000	vers espèce
credit	2021-02-02	5000	vers espèce
Total des operations de crédit		14000	

B) Partie Mapping Objet XML avec Jax Binding

1) Classe Operation




m.xml (oxm)
Operation.java
TypeDescription.java
TypeOperation.java
Releve.java
4 usages
public Operation(TypeOperation type, Date date, double montant, TypeDescription description) {
this.type = type;
this.date = date;
this.montant = montant;
this.description = description;
}
no usages
public TypeOperation getType() {
return type;
}
no usages
public void setType(TypeOperation type) {
this.type = type;
}
no usages
public Date getDate() {
return date;
}
no usages
public void setDate(Date date) {
this.date = date;
}


pom.xml (oxm)
Operation.java
TypeDescription.java
TypeOpe
public double getMontant() {
return montant;
}
no usages
public void setMontant(double montant) {
this.montant = montant;
}
no usages
public TypeDescription getDescription() {
return description;
}
no usages
public void setDescription(TypeDescription description) {
this.description = description;
}
@Override
public String toString() {
return "Operation{" +
"type=" + type +
", date=" + date +
", montant=" + montant +
", description=" + description +
'}';
}
}

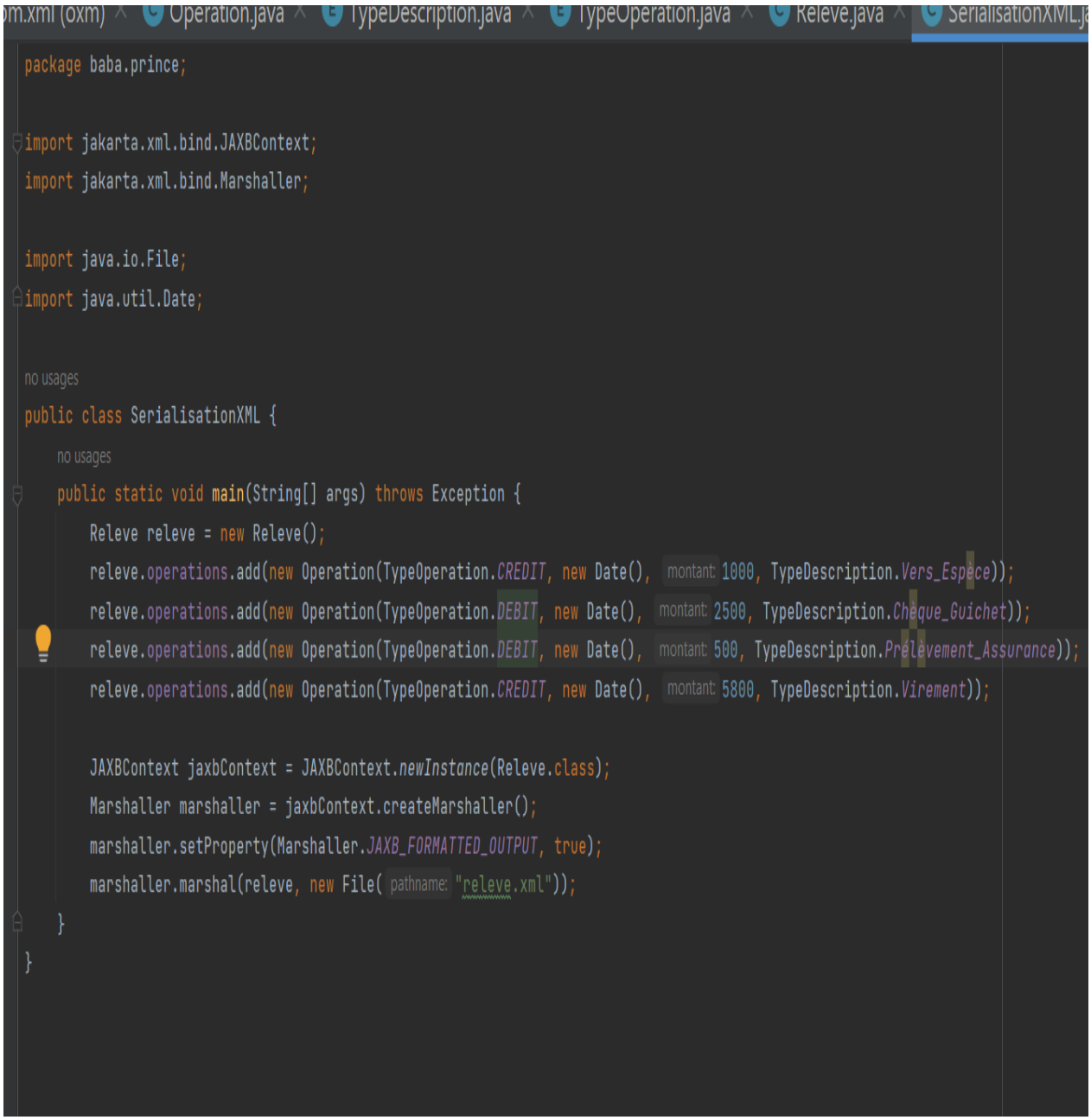
2) Classe Releve

pom.xml (oxm) | Operation.java | TypeDescription.java

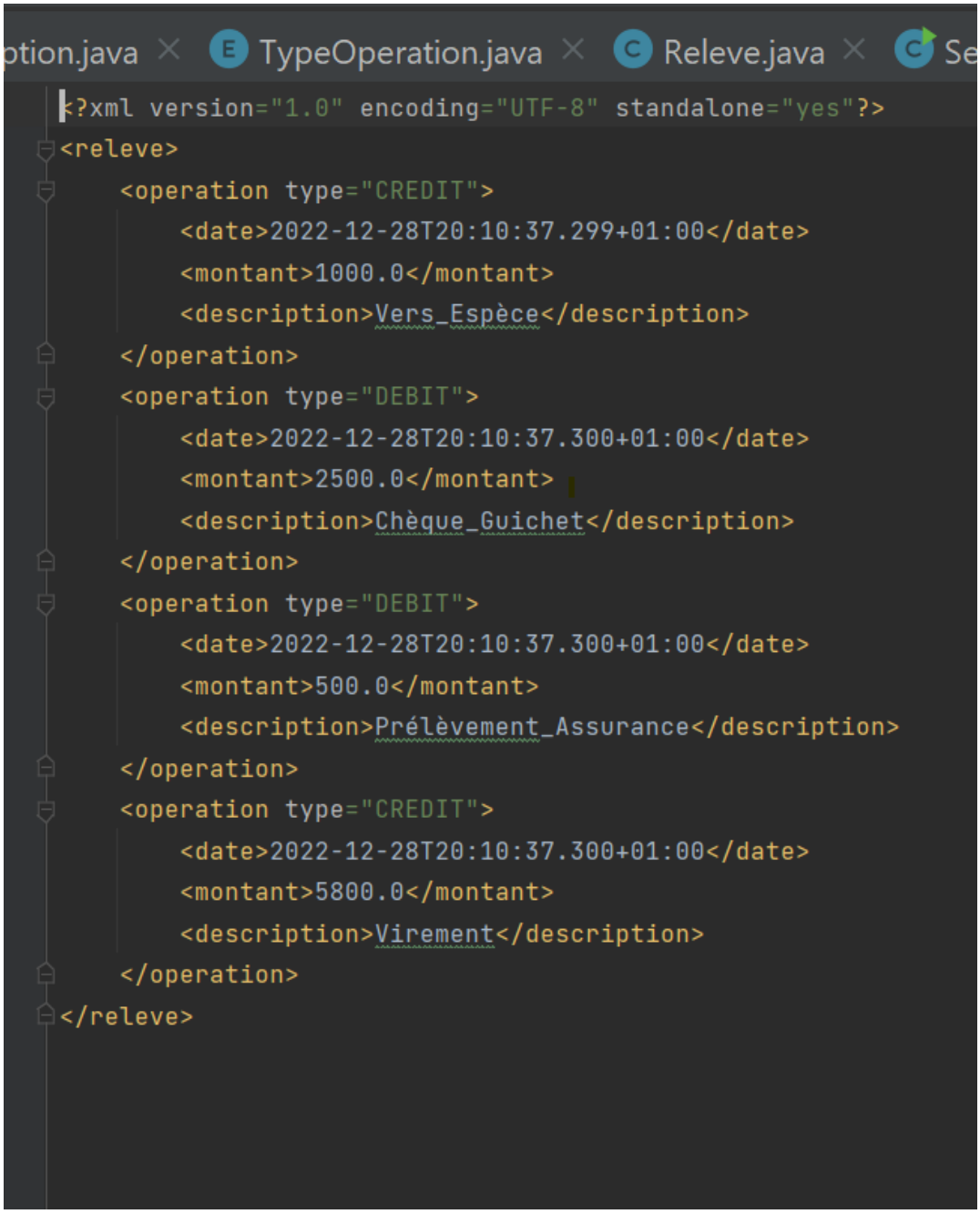
```java
package baba.prince;

import jakarta.xml.bind.annotation.XmlElement;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.util.ArrayList;
import java.util.List;

7 usages
@XmlRootElement
public class Releve {
    5 usages
    @XmlElement(name = "operation")
    public List<Operation> operations = new ArrayList<>();
}
```

3) Classe Serialisation

Cette classe permet de créer un objet « Releve » contenant une liste d'opérations et de sérialiser ces données dans un fichier XML



- **Fichier XML contenant les données de la sérialisation**



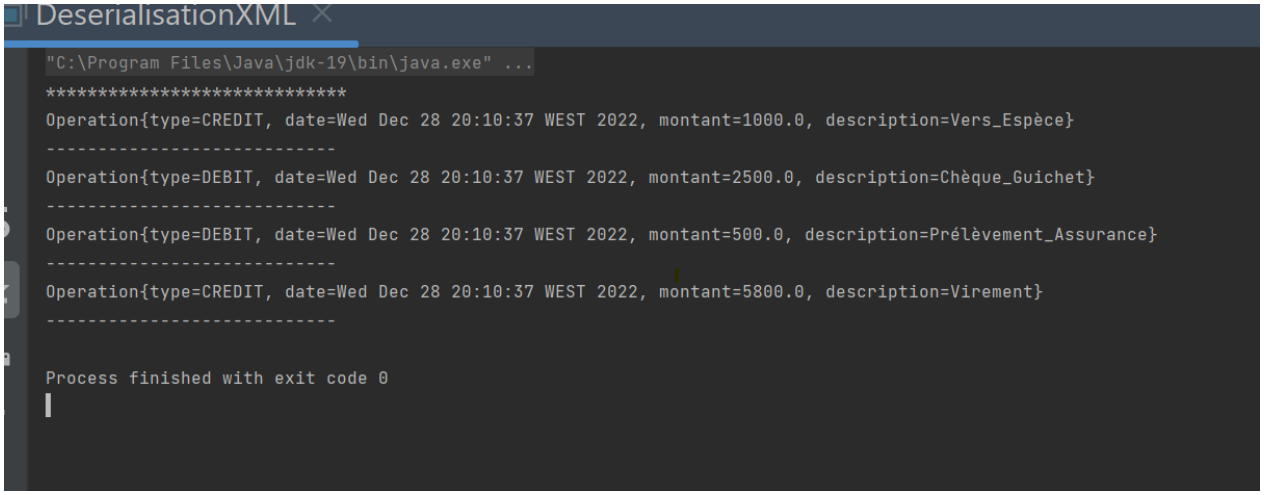
4) Classe Deserialisation

Cette classe permet de lire et d'afficher les données du releve du fichier XML généré précédemment.

TypeDescription.java | TypeOperation.java | Releve.java | SerialisationXML.java

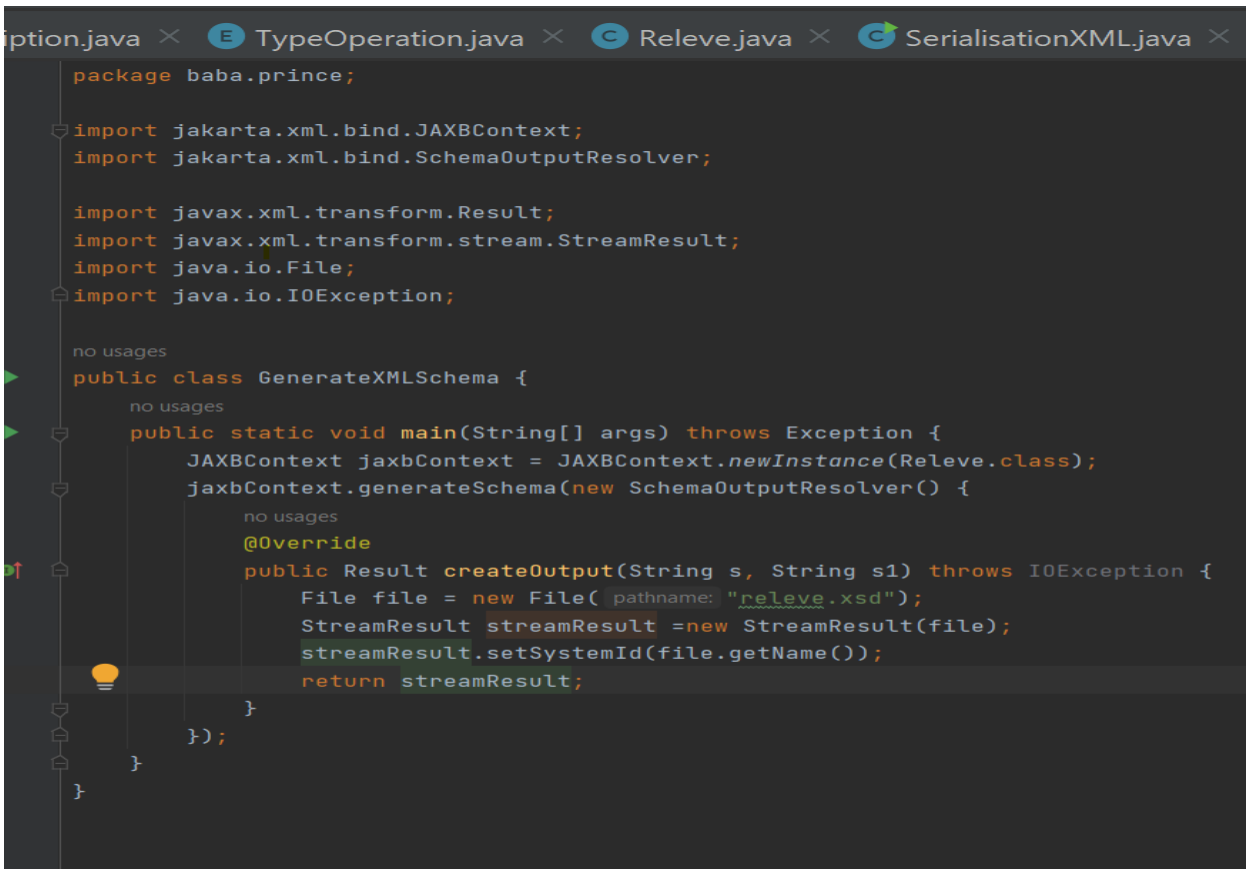
```java
package baba.prince;

import jakarta.xml.bind.JAXBContext;
import jakarta.xml.bind.Unmarshaller;

import java.io.File;

no usages
public class DeserialisationXML {
    no usages
    public static void main(String[] args) throws Exception {
        JAXBContext jaxbContext = JAXBContext.newInstance(Releve.class);
        Unmarshaller unmarshaller = jaxbContext.createUnmarshaller();
        Releve releve = (Releve) unmarshaller.unmarshal(new File( pathname: "releve.xml"));
        System.out.println("**************************");
        for (Operation op : releve.operations) {
            System.out.println(op.toString());
            System.out.println("--------------------------");
        }
    }
}
```

- **Résultat de la déserialisation**



5) Classe GenerateXMLShcema

Cette classe permet de générer le schema XML representant la structure d'un releve.



- **Schéma XML representant la structure d'un releve**

TypeOperation.java | Releve.java | SerialisationXML.java | releve.xm

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="operation" type="operation"/>

  <xs:element name="releve" type="releve"/>

  <xs:complexType name="releve">
    <xs:sequence>
      <xs:element ref="operation" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="operation">
    <xs:sequence>
      <xs:element name="date" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="montant" type="xs:double"/>
      <xs:element name="description" type="typeDescription" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="type" type="typeOperation"/>
  </xs:complexType>

  <xs:simpleType name="typeOperation">
    <xs:restriction base="xs:string">
      <xs:enumeration value="CREDIT"/>
      <xs:enumeration value="DEBIT"/>
    </xs:restriction>
  </xs:simpleType>
```

C) Partie C Web Services

1) Web Services

- **Classe Operation**

```java
package baba.prince;

import jakarta.xml.bind.annotation.XmlAccessType;
import jakarta.xml.bind.annotation.XmlAccessorType;
import jakarta.xml.bind.annotation.XmlAttribute;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.util.Date;

7 usages
public class Operation {
    4 usages
    private TypeOperation type;
    4 usages
    private Date date;
    4 usages
    private double montant;
    4 usages
    private TypeDescription description;

    no usages
    public Operation() {
    }

    4 usages
    public Operation(TypeOperation type, Date date, double montant, TypeDescription description) {
        this.type = type;
        this.date = date;
        this.montant = montant;
        this.description = description;
    }
```

```java
public TypeOperation getType() {
    return type;
}

no usages
public void setType(TypeOperation type) {
    this.type = type;
}

no usages
public Date getDate() {
    return date;
}

no usages
public void setDate(Date date) {
    this.date = date;
}

no usages
public double getMontant() {
    return montant;
}

no usages
public void setMontant(double montant) {
    this.montant = montant;
}
```

```
no usages
public TypeDescription getDescription() {
    return description;
}

no usages
public void setDescription(TypeDescription description) {
    this.description = description;
}

@Override
public String toString() {
    return "Operation{" +
            "type=" + type +
            ", date=" + date +
            ", montant=" + montant +
            ", description=" + description +
            '}';
}
```

- **Classe Releve**

```java
package baba.prince;

import jakarta.jws.WebService;
import jakarta.xml.bind.annotation.XmlElement;
import jakarta.xml.bind.annotation.XmlRootElement;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;

17 usages
public class Releve {
    3 usages
    private long rib;
    6 usages
    private List<Operation> operations = new ArrayList<>();
    3 usages
    private Date dateReleve;
    3 usages
    private double solde;

    10 usages
    public Releve(long rib, Date dateReleve, double solde) {
        this.rib = rib;
        operations.add(new Operation(TypeOperation.CREDIT, new Date(), montant: Math.random()*9800, TypeDescription.Vers_Espece));
        operations.add(new Operation(TypeOperation.DEBIT, new Date(), montant: Math.random()*5600, TypeDescription.Virement));
        operations.add(new Operation(TypeOperation.CREDIT, new Date(), montant: Math.random()*3500, TypeDescription.Cheque_Guichet));
        operations.add(new Operation(TypeOperation.DEBIT, new Date(), montant: Math.random()*100, TypeDescription.Prelevement_Assurance));
        this.dateReleve = dateReleve;
        this.solde = solde;
    }
```

```java
public long getRib() {
    return rib;
}

no usages
public void setRib(long rib) {
    this.rib = rib;
}

no usages
public List<Operation> getOperations() {
    return operations;
}

no usages
public void setOperations(List<Operation> operations) {
    this.operations = operations;
}

1 usage
public Date getDateReleve() {
    return dateReleve;
}

no usages
public void setDateReleve(Date dateReleve) {
    this.dateReleve = dateReleve;
}

1 usage
public double getSolde() {
    return solde;
}
```

- **Classe ReleveService**

Cette classe un Web services basé sur JaxWS qui permet de consulter un relevé

```java
package baba.prince;

import jakarta.jws.WebMethod;
import jakarta.jws.WebParam;
import jakarta.jws.WebService;

import java.util.Date;
import java.util.List;

2 usages
@WebService(serviceName = "ReleveWS")
public class ReleveService {
    no usages
    @WebMethod
    public Releve getReleve(@WebParam(name = "rib") long rib) {
        return new Releve(rib, new Date(), solde: Math.random()*43000);
    }

    no usages
    @WebMethod
    public List<Releve> listReleve() {
        return List.of(
                new Releve( rib: 1, new Date(), solde: Math.random()*2000),
                new Releve( rib: 2, new Date(), solde: Math.random()*87000),
                new Releve( rib: 3, new Date(), solde: Math.random()*2400),
                new Releve( rib: 4, new Date(), solde: Math.random()*9400),
                new Releve( rib: 5, new Date(), solde: Math.random()*8600),
                new Releve( rib: 6, new Date(), solde: Math.random()*4800)
        );
    }

}
```

2) Serveur JaxWS

```java
package server;

import baba.prince.ReleveService;
import jakarta.xml.ws.Endpoint;

no usages
public class ServerJWS {
    no usages
    public static void main(String[] args) {
        Endpoint.publish( address: "http://0.0.0.0:9191/", new ReleveService());
        System.out.println("Web Service deployé sur http://0.0.0.0:9191/");
    }
}
```

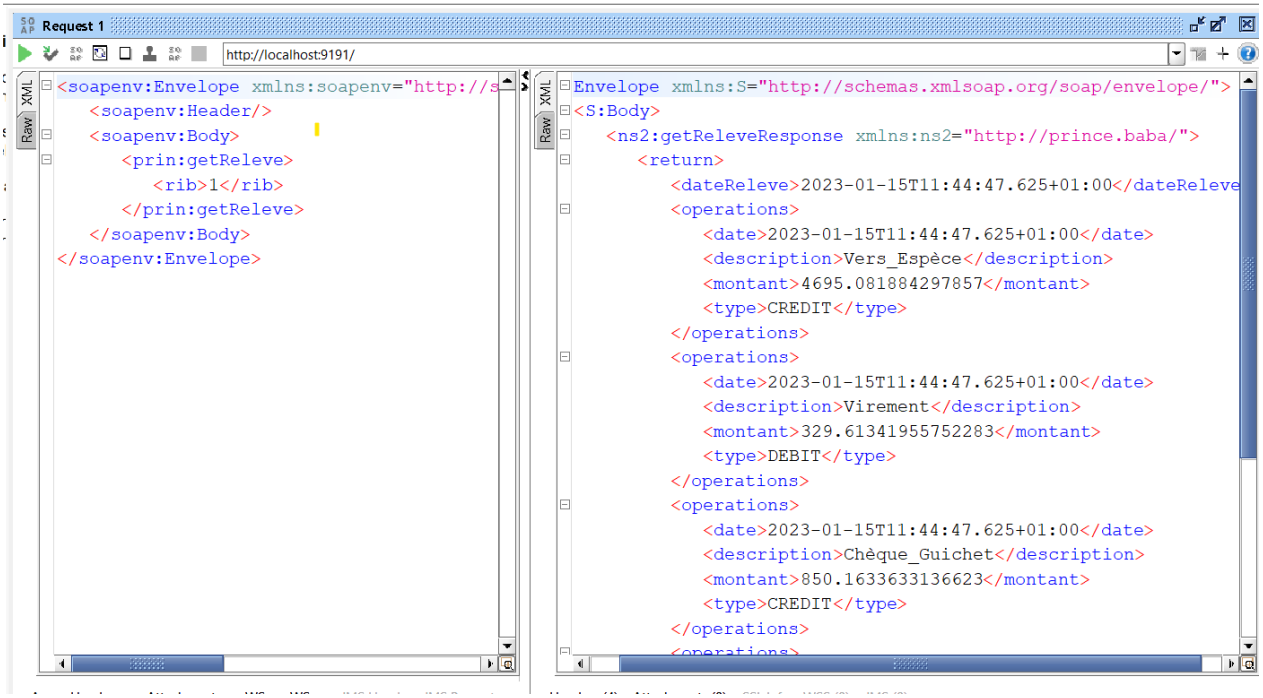
3) WSDL

This XML file does not appear to have any style information associated with it. The document tree is shown below.

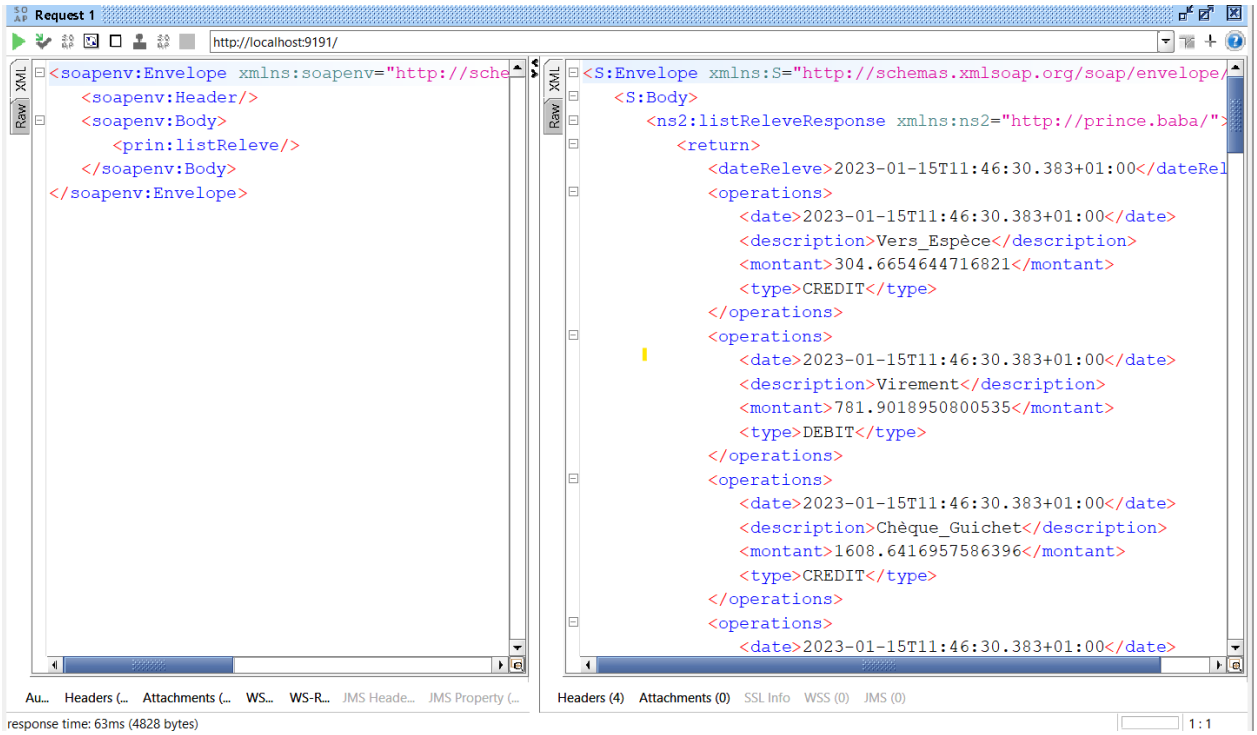
```
<!--  Published by XML-WS Runtime (https://github.com/eclipse-ee4j/metro-jax-ws). Runtime's version is XML-WS Runtime 4.0.0 git-revision#129f787.  -->
<!--  Generated by XML-WS Runtime (https://github.com/eclipse-ee4j/metro-jax-ws). Runtime's version is XML-WS Runtime 4.0.0 git-revision#129f787.  -->
<definitions xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd" xmlns:wsp="http://www.w3.org/ns/ws-policy"
 xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:tns="http://prince.baba/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http://prince.baba/" name="ReleveWS">
  <types>
    <xsd:schema>
      <xsd:import namespace="http://prince.baba/" schemaLocation="http://localhost:9191/?xsd=1"/>
    </xsd:schema>
  </types>
  <message name="getReleve">
    <part name="parameters" element="tns:getReleve"/>
  </message>
  <message name="getReleveResponse">
    <part name="parameters" element="tns:getReleveResponse"/>
  </message>
  <message name="listReleve">
    <part name="parameters" element="tns:listReleve"/>
  </message>
  <message name="listReleveResponse">
    <part name="parameters" element="tns:listReleveResponse"/>
  </message>
  <portType name="ReleveService">
    <operation name="getReleve">
      <input wsam:Action="http://prince.baba/ReleveService/getReleveRequest" message="tns:getReleve"/>
      <output wsam:Action="http://prince.baba/ReleveService/getReleveResponse" message="tns:getReleveResponse"/>
    </operation>
    <operation name="listReleve">
      <input wsam:Action="http://prince.baba/ReleveService/listReleveRequest" message="tns:listReleve"/>
      <output wsam:Action="http://prince.baba/ReleveService/listReleveResponse" message="tns:listReleveResponse"/>
    </operation>
  </portType>
  <binding name="ReleveServicePortBinding" type="tns:ReleveService">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="document"/>
    <operation name="getReleve">
      <soap:operation soapAction=""/>
      <input>
        <soap:body use="literal"/>
      </input>
```

4) Test des méthodes avec SoapUI

- **Method getReleve**



- **Method listReleve**



5) Client SOAP Java

- **Classe ClientWS**

```java
import proxy.Operation;
import proxy.Releve;
import proxy.ReleveService;
import proxy.ReleveWS;

no usages
public class ClientWS {
    no usages
    public static void main(String[] args) {
        ReleveService stub = new ReleveWS().getReleveServicePort();
        System.out.println("rib : " + stub.getReleve( rib: 1).getRib());
        System.out.println("date releve : " + stub.getReleve( rib: 1).getDateReleve());
        System.out.println("solde : " + stub.getReleve( rib: 1).getSolde());
        System.out.println("");
    }
}
```

- **Resultat de la requête**

```
ClientWS
"C:\Program Files\Java\jdk-19\bin\java.exe" ...
rib : 1
date releve : 2023-01-15T11:48:19.693+01:00
solde : 35927.474873964806


Process finished with exit code 0
```