



Instituto Politécnico
de Viana do Castelo

ipvc

Instituto Politécnico
de Viana do Castelo



Escola Superior
de Tecnologia e Gestão

Versão 1 – Documento em Atualização

Integração de Sistemas e Tecnologias

Interoperabilidade de Sistemas e Tecnologias

Tutorial 9 – WebServices, XML, SOAP, Java, PHP..

Escola Superior de Tecnologia e Gestão
Instituto Politécnico de Viana do Castelo
Braga, Portugal

Jorge Ribeiro

jribeiro@estg.ipvc.pt



- Exercício – DOM (simple.xml, books.xml, medsamp.xml)
- Exercício – SAX (simple.xml, books.xml, medsamp.xml, note.xml)
- Exercício – JDOM (books.xml, medsamp.xml, note.xml)
- Exercício – JAXP (simple.xml, books.xml, medsamp.xml)
- Exercício – JAXB (po.xml)
- SOAP – JAVA:
 - Exemplo nº1 – Adicionar dois números,
 - Exemplo nº2 – Devolver o ficheiro Books.xml para leitura
 - Exemplo nº3 – Devolver dados da base de dados
- SOAP – PHP:
 - Exemplo nº1: Exemplo dado pelo sitepoint.com
 - Exemplo nº2: Devolver o ficheiro Books.xml para leitura
 - Exemplo nº3: Devolver registos da base de dados em XML
- REST – JAVA
 - Exemplo nº1: Devolver o ficheiro Books.xml para leitura
 - Exemplo nº2: Acesso à base de dados e devolução de registos em XML
- REST – PHP
 - Exemplo nº1: Acesso à base de dados e devolução de registos em XML
 - Exemplo nº2: Enviar conteúdo do ficheiro Books.xml
- JAVA – SOAP – DOM, SAX, JDOM, JAXP, JAXB
- JAVA – REST – DOM, SAX, JDOM, JAXP, JAXB
- PHP – SOAP – DOM, SAX, JDOM, JAXP, JAXB
- PHP – REST – DOM, SAX, JDOM, JAXP, REST - JAXB
- PHP – SOAP – JDOM com FileUpload



Ambiente de Desenvolvimento para o Tutorial 9

Netbeans IDE 8.2 ALL (JAVA WebServices)

Glassfish (Java WebServices – Servidor WEB)

Visual Studio Code (PHP WebServices)

WAMP – 3.1.9 64bit (PHP WebServices)

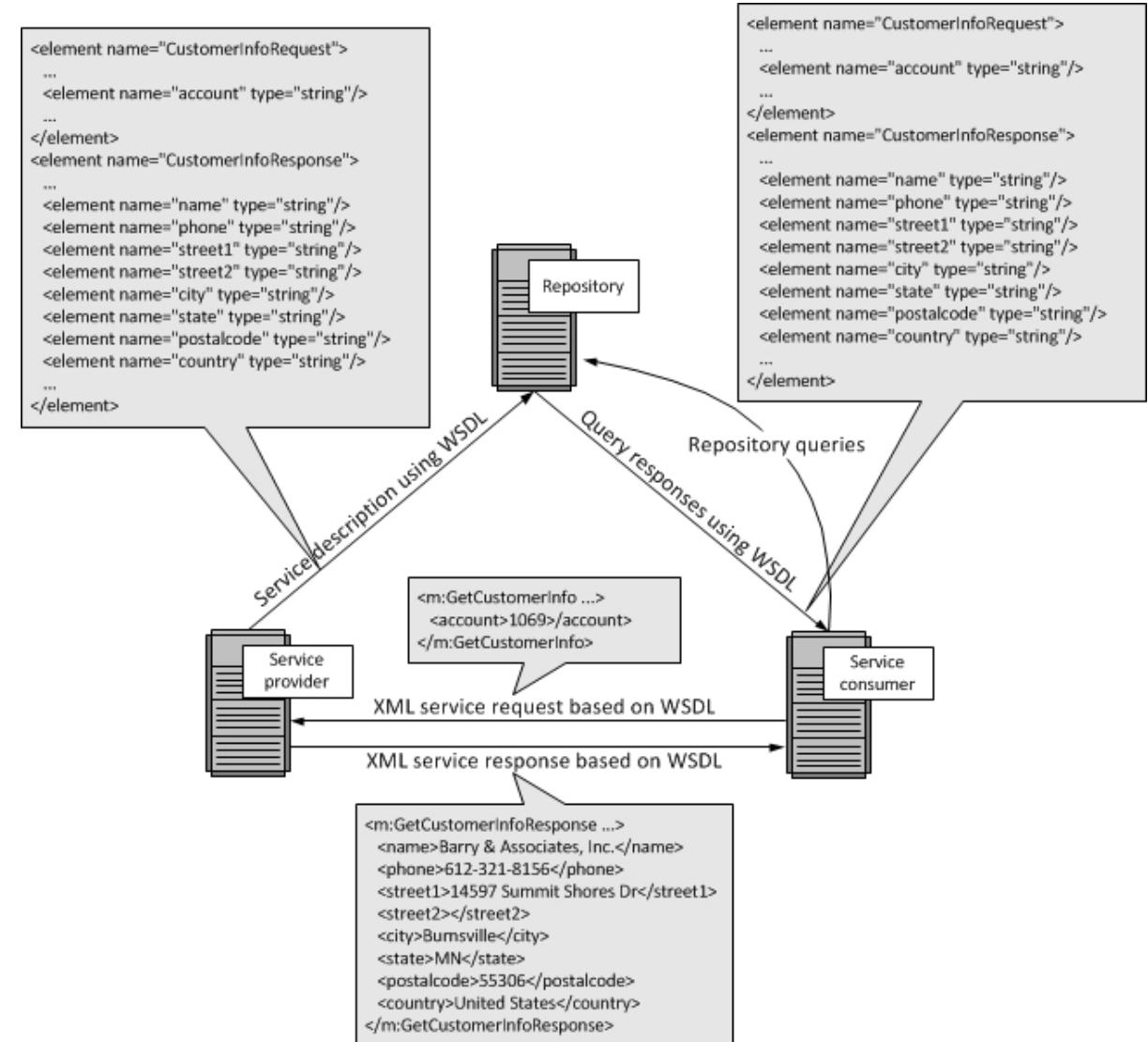
- **Apache 2.4.39**
- **Mysql 5.7.26**
- **PHP 7.2.18**

Testes:

- **SOAP UI**
- **POSTMAN**
- **BROWSER**

SOAP

SOAP was originally part of the specification that included the Web Services Description Language (WSDL) and Universal Description, Discovery, and Integration (UDDI). It is used now without WSDL and UDDI. Instead of the discovery process described in the History of the Web Services Specification section below, SOAP messages are hard-coded or generated without the use of a repository. The interaction is illustrated in the figure below.





SOAP – JAVA

Exemplo nº1 – Adicionar dois números

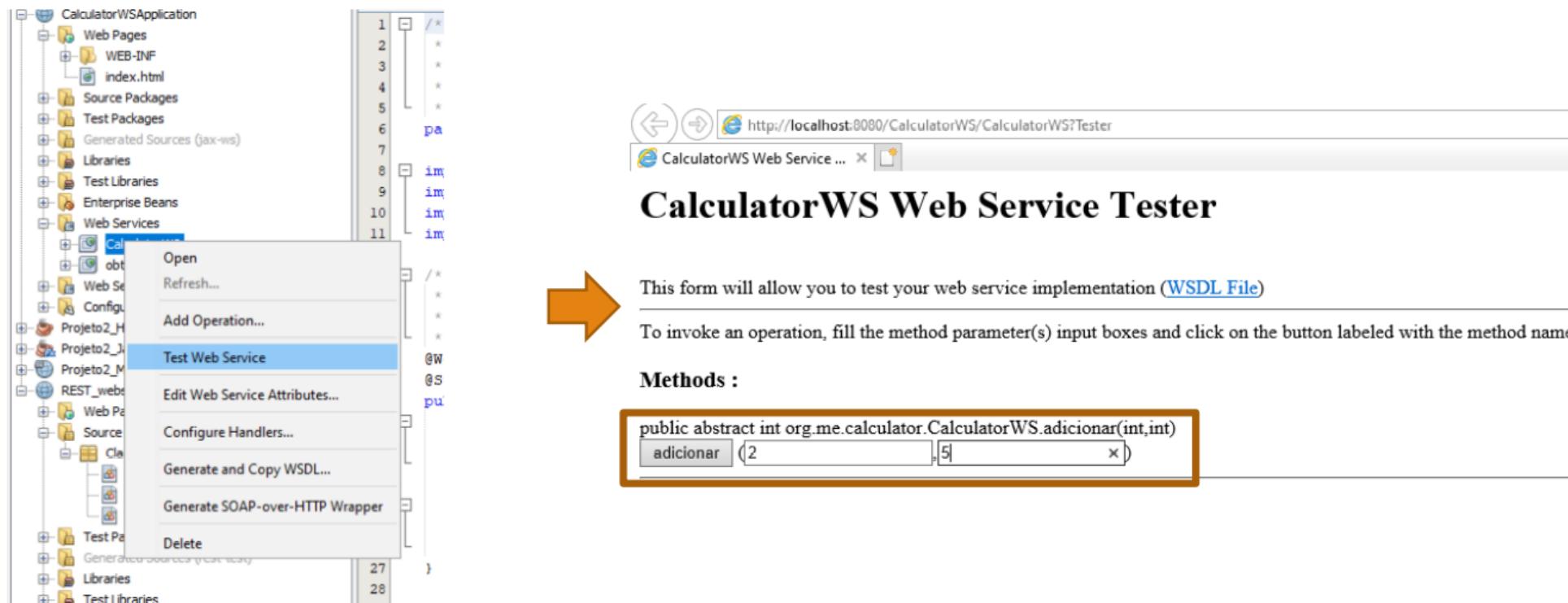
Código

```
@WebService(serviceName = "CalculatorWS")
@Stateless()
public class CalculatorWS {
    /**
     * Web service operation
     */
    @WebMethod(operationName = "adicionar")
    public int adicionar(@WebParam(name = "a") int a, @WebParam(name = "b") int b) {
        return a+b;
    }
}
```

SOAP – JAVA

Exemplo nº1 – Adicionar dois números

Testes – Browser (Opção “Test Web Service” do Netbeans)





SOAP – JAVA

Exemplo nº1 – Adicionar dois números

Testes – Browser (Opção “Test Web Service” do Netbeans)

The screenshot shows the 'CalculatorWS/CalculatorWSTester' page in a browser. The URL is `http://localhost:8080/CalculatorWS/CalculatorWSTester`. A table titled 'Method parameter(s)' contains two rows: 'int' with value '2' and 'int' with value '5'. To the right, a box labeled 'Valor devolvido pelo Web Service' contains the result 'int : "7"'. Below this, the 'Method returned' section also shows 'int : "7"'. The 'SOAP Request' and 'SOAP Response' sections at the bottom show the XML code for the SOAP message.

Method parameter(s)

Type	Value
int	2
int	5

Valor devolvido pelo Web Service

Method returned

int : "7"

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/>
<S:Body>
<ns2:adicionar xmlns:ns2="http://calculator.me.org/">
<a>2</a>
<b>5</b>
</ns2:adicionar>
</S:Body>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/>
<S:Body>
<ns2:adicionarResponse xmlns:ns2="http://calculator.me.org/">
<return>7</return>
</ns2:adicionarResponse>
</S:Body>
</S:Envelope>
```



SOAP – JAVA

Exemplo nº1 – Adicionar dois números

Testes – Browser (Opção “Test Web Service” do Netbeans)

The screenshot shows the 'Method invocation trace' tab in the Netbeans IDE. A method named 'adicionar' is selected. In the 'Method parameter(s)' section, there is a table with two rows:

Type	Value
int	2
int	5

In the 'Method returned' section, the value 'int : "7"' is displayed. Below this, the 'SOAP Request' section contains the XML code for the SOAP message:

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<S:Header>
<S:Body>
<ns2:adicionar xmlns:ns2="http://calculator.me.org/">
<a>2</a>
<b>5</b>
</ns2:adicionar>
</S:Body>
</S:Envelope>
```

Formato do Pedido SOAP

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<S:Header>
<S:Body>
<ns2:adicionar xmlns:ns2="http://calculator.me.org/">
<a>2</a>
<b>5</b>
</ns2:adicionar>
</S:Body>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<S:Header>
<S:Body>
<ns2:adicionarResponse xmlns:ns2="http://calculator.me.org/">
<return>7</return>
</ns2:adicionarResponse>
</S:Body>
</S:Envelope>
```



SOAP – JAVA

Exemplo nº1 – Adicionar dois números

Testes – Browser (Opção “Test Web Service” do Netbeans)

The screenshot shows the 'CalculatorWS?Tester' window in Netbeans. The 'Method invocation trace' tab is selected. The 'adicionar Method invocation' section shows the following parameters:

Type	Value
int	2
int	5

The 'Method returned' section shows the result: int : "7".

The 'SOAP Request' section displays the XML envelope sent to the server:

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/>
<S:Body>
<ns2:adicionar xmlns:ns2="http://calculator.me.org/">
<a>2</a>
<b>5</b>
</ns2:adicionar>
</S:Body>
</S:Envelope>
```

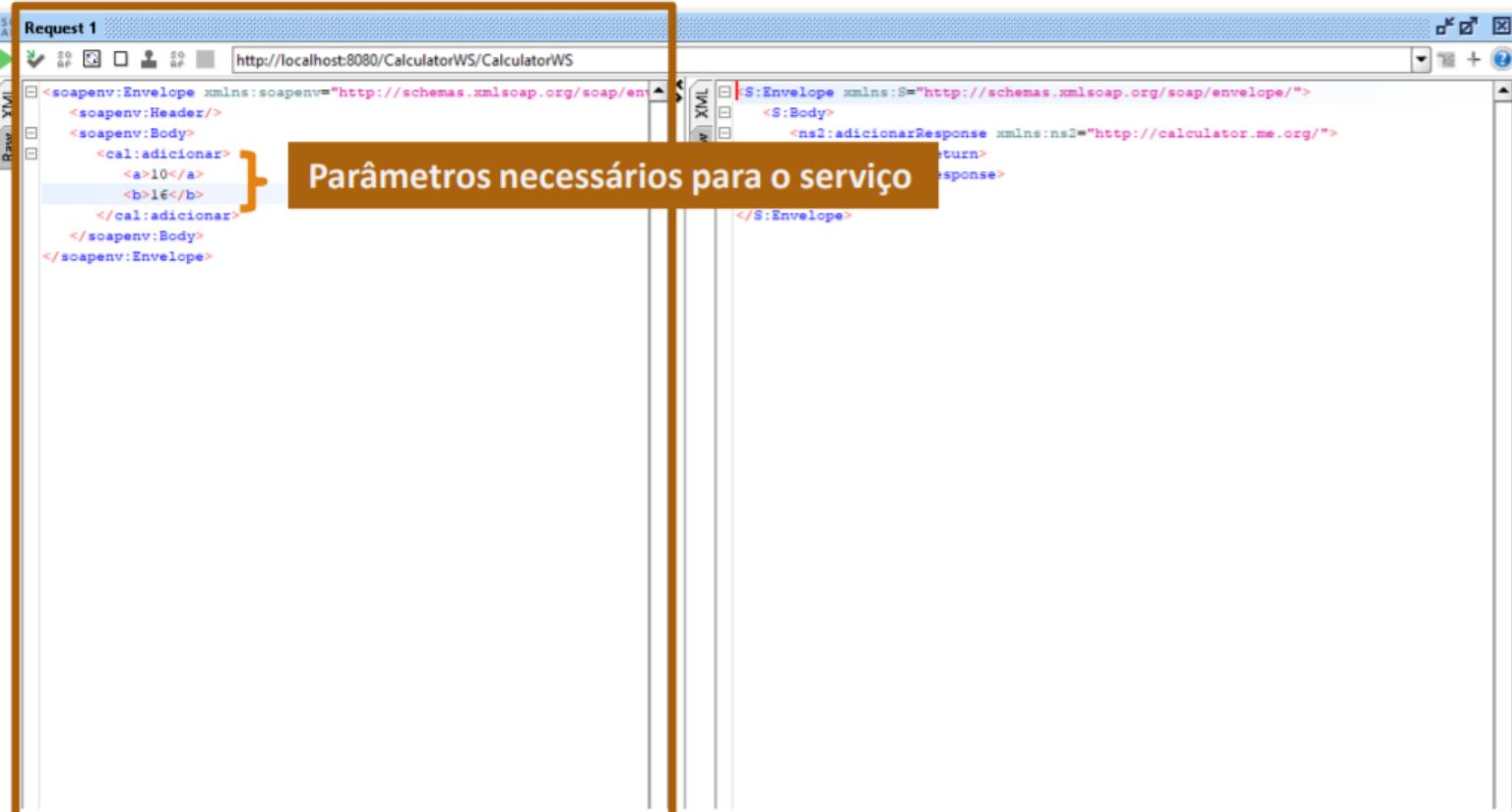
The 'Resposta SOAP do servidor' (Server's SOAP Response) section shows the XML envelope received from the server:

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/>
<S:Body>
<ns2:adicionarResponse xmlns:ns2="http://calculator.me.org/">
<return>7</return>
</ns2:adicionarResponse>
</S:Body>
</S:Envelope>
```

SOAP – JAVA

Exemplo nº1 – Adicionar dois números

Testes – SoapUI



The screenshot shows the SoapUI interface with a request labeled "Request 1". The URL is "http://localhost:8080/CalculatorWS/CalculatorWS". The request XML is:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <cal:adicionar>
      <a>10</a>
      <b>16</b>
    </cal:adicionar>
  </soapenv:Body>
</soapenv:Envelope>
```

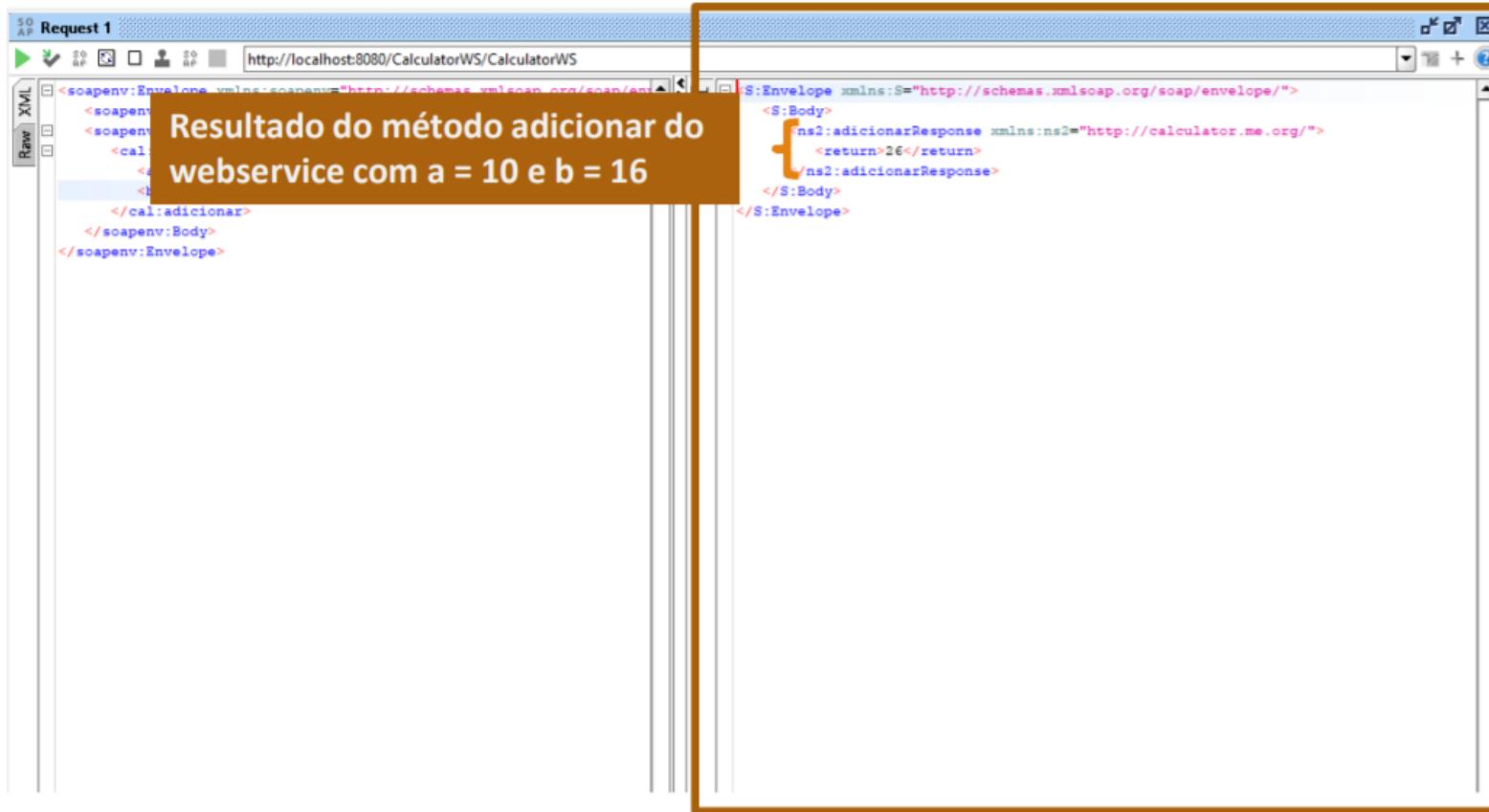
A callout box points to the parameters "a" and "b" with the text "Parâmetros necessários para o serviço".

Pedido SOAP

SOAP – JAVA

Exemplo nº1 – Adicionar dois números

Testes – SoapUI



The screenshot shows the SoapUI interface with a request and response pane. The request pane contains a SOAP message to add two numbers (a=10, b=16). The response pane shows the XML response, which includes a namespace declaration and a return value of 26.

Resultado do método adicionar do
webservice com a = 10 e b = 16

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:adicionarResponse xmlns:ns1="http://calculator.me.org/">
      <return>26</return>
    </ns1:adicionarResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Resultado SOAP



SOAP – JAVA

Exemplo nº2 – Devolver o ficheiro Books.xml para leitura

Código

```
@WebService(serviceName = "obtemXML")
public class obtemXML {

    private final String caminhoBooksXml = "D:\\IPVC\\5ºS. - Integração de Sistemas\\Trabalhos Práticos\\TPI - Webservices SOAP e REST\\JAVA\\REST_webservice\\books.xml";

    /**
     * This is a sample web service operation
     */
    @WebMethod(operationName = "obtemXML")
    public String obtemXML() {
        try {
            byte[] encoded = Files.readAllBytes(Paths.get(caminhoBooksXml));
            String retorno = new String(encoded, "utf-8");

            return retorno;
        } catch (IOException ex) {
            return "ERRO: " + ex.getMessage();
        }
    }
}
```



SOAP – JAVA

Exemplo nº2 – Devolver o ficheiro Books.xml para leitura

Testes – Browser (Opção “Test Web Service” do Netbeans)

The screenshot shows the Netbeans IDE interface. On the left, the project tree displays a Java web application structure with a 'CalculatorWS' module containing an 'obtemXML' operation. A context menu is open over the 'obtemXML' node, with the 'Test Web Service' option highlighted. To the right, the 'obtemXML Web Service Tester' window is open, showing the URL <http://localhost:8080/CalculatorWSApplication/obtemXML?Tester>. The tester interface includes a large orange arrow pointing from the context menu towards the tester window, indicating the selection of the 'Test Web Service' option.

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

```
public abstract java.lang.String org.me.calculator.ObtemXML.obtemXML()
```

obtemXML



SOAP – JAVA

Exemplo nº2 – Devolver o ficheiro Books.xml para leitura

Testes – Browser (Opção “Test Web Service” do Netbeans)

The screenshot shows a browser window with the URL <http://localhost:8080/CalculatorWS/CalculatorWSApplication/ultimoXML/Teste>. The page title is "obtemXML Method invocation". The main content area is titled "Valor devolvido pelo Web Service" and contains a large block of XML code representing the response from the web service.

Method parameter(s)

Type	Value

Method returned

```
java.lang.String: "<?xml version='1.0'?><catalog> <book id='bk101'> <author>Gambardella, Matthew</author> <title>XML Developer's Guide</title> <genre>Computer</genre> <price>44.95</price> <publish_date>2000-10-01</publish_date> <description>An in-depth look at creating applications with XML</description> <book> <book id='bk102'> <author>Ralls, Kim</author> <title>Midnight Rain</title> <genre>Fantasy</genre> <price>5.95</price> <publish_date>2000-12-16</publish_date> <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description> <book> <book id='bk103'> <author>Corets, Eva</author> <title>Maeve Ascendant</title> <genre>Fantasy</genre> <price>5.95</price> <publish_date>2000-11-17</publish_date> <description>After the collapse of a nanotechnology society in England, the young survivors help to found the new society.</description> <book> <book id='bk104'> <author>Corets, Eva</author> <title>Oberon's Legacy</title> <genre>Fantasy</genre> <price>5.95</price> <publish_date>2001-03-18</publish_date> <description>In post-apocalyptic England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Maeve Ascendant.</description> <book> <book id='bk105'> <author>Corets, Eva</author> <title>The Sundered Grail</title> <genre>Fantasy</genre> <price>5.95</price> <publish_date>2001-09-10</publish_date> <description>The two daughters of Maeve, half-sisters, battle one another for control of England. Sequel to Oberon's Legacy.</description> <book> <book id='bk106'> <author>Randall, Cynthia</author> <title>Love Birds</title> <genre>Romance</genre> <price>4.95</price> <publish_date>2000-09-02</publish_date> <description>When Carla meets Paul at an ornithology conference, tempers fly as feathers get ruffled.</description> <book> <book id='bk107'> <author>Tharman, Paula</author> <title>Splash Splash</title> <genre>Romance</genre> <price>4.95</price> <publish_date>2000-11-02</publish_date> <description>A deep sea diver finds true love twenty thousand leagues beneath the sea.</description> <book> <book id='bk108'> <author>Knauf, Stefan</author> <title>Creepy Crawlies</title> <genre>Horror</genre> <price>4.95</price> <publish_date>2000-12-06</publish_date> <description>An anthology of horror stories about roaches, centipedes, scorpions and other insects.</description> <book> <book id='bk109'> <author>Kress, Peter</author> <title>Paradox Lost</title> <genre>Science Fiction</genre> <price>6.95</price> <publish_date>2000-11-02</publish_date> <description>After an inadvertent trip through a Heisenberg Uncertainty Device, James Salway discovers the problems of being quantum.</description> <book> <book id='bk110'> <author>O'Brien, Tim</author> <title>Microsoft .NET: The Programming Bible</title> <genre>Computer</genre> <price>36.95</price> <publish_date>2000-12-09</publish_date> <description>Microsoft's .NET initiative is explored in detail in this deep programmer's reference.</description> <book> <book id='bk111'> <author>O'Brien, Tim</author> <title>MSXML3: A Comprehensive Guide</title> <genre>Computer</genre> <price>36.95</price> <publish_date>2000-12-01</publish_date> <description>The Microsoft MSXML3 parser is covered in detail, with attention to XML DOM interfaces, XSLT processing, SAX and more.</description> <book> <book id='bk112'> <author>Giles, Mike</author> <title>Visual Studio 7: A Comprehensive Guide</title> <genre>Computer</genre> <price>49.95</price> <publish_date>2001-04-16</publish_date> <description>Microsoft Visual Studio 7 is explored in depth, looking at how Visual Basic, Visual C++, C#, and ASP+ are integrated into a comprehensive development environment.</description> </book> </catalog>"
```

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><?soap:Envelope xmlns:xsi="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"?>
<?soap:Header?>
<?soap:Body?>
    <ns1:obtemXML xmlns:ns1="http://calculator.ws.xml"/>
</?soap:Body?>
</?soap:Envelope?>
```

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><?soap:Envelope xmlns:xsi="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"?>
<?soap:Header?>
<?soap:Body?>
    <ns1:obtemXMLResponse xmlns:ns1="http://calculator.ws.xml"/>
    <returns><?xml version="1.0"?>
<catalog>
    <book id='bk101'>
        <author>Gambardella, Matthew</author>
        <title>XML Developer's Guide</title>
        <genre>Computer</genre>
    </book>
</catalog>
</returns>
</?soap:Body?>
</?soap:Envelope?>
```

SOAP – JAVA

Exemplo nº2 – Devolver o ficheiro Books.xml para leitura

Testes – Browser (Opção “Test Web Service” do Netbeans)

The screenshot shows a browser window with the URL <http://localhost:3000/CalculatorWS/Application/obtemXML/Teste>. The page title is "obtemXML Method invocation". The content area displays the XML response from the server, which includes details about books like "XML Developer's Guide", "Midnight Rain", "Maeve Ascendant", etc., along with their descriptions and publication dates.

Method parameter(s)

Type	Value
------	-------

Method returned

```
java.lang.String : <?xml version="1.0"?> <catalog> <book id="bk101"> <author>Gambardella, Matthew</author> <title>XML Developer's Guide</title> <genre>Computer</genre> <price>44.95</price> <publish_date>2000-10-01</publish_date> <description>An in-depth look at creating applications with XML.</description> <book id="bk102"> <author>Ralls, Kim</author> <title>Midnight Rain</title> <genre>Fantasy</genre> <price>5.95</price> <publish_date>2000-12-16</publish_date> <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description> <book id="bk103"> <author>Corets, Eva</author> <title>Maeve Ascendant</title> <genre>Fantasy</genre> <price>5.95</price> <publish_date>2000-11-01</publish_date> <description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description> <book id="bk104"> <author>Corets, Eva</author> <title>Oberon's Legacy</title> <genre>Fantasy</genre> <price>5.95</price> <publish_date>2001-03-10</publish_date> <description>In post-apocalyptic England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Maeve Ascendant.</description> <book id="bk105"> <author>Corets, Eva</author> <title>The Sundered Grail</title> <genre>Fantasy</genre> <price>5.95</price> <publish_date>2001-09-10</publish_date> <description>The two daughters of Maeve, half-sisters, battle one another for control of England. Sequel to Oberon's Legacy.</description> <book id="bk106"> <author>Randall, Cynthia</author> <title>Lover Birds</title> <genre>Romance</genre> <price>4.95</price> <publish_date>2000-09-02</publish_date> <description>When Carla meets an ornithology conference, tempers fly as feathers get ruffled.</description> <book id="bk107"> <author>Thurman, Paula</author> <title>Splish Splash</title> <genre>Romance</genre> <price>4.95</price> <publish_date>2000-11-02</publish_date> <description>A deep sea diver finds true love twenty thousand leagues beneath the sea.</description> <book id="bk108"> <author>Knorr, Stefan</author> <title>Creepy Crawlies</title> <genre>Horror</genre> <price>4.95</price> <publish_date>2000-12-06</publish_date> <description>An anthology of horror stories about roaches, centipedes, scorpions and other insects.</description> <book id="bk109"> <author>Kress, Peter</author> <title>Paradox Lost</title> <genre>Science Fiction</genre> <price>6.95</price> <publish_date>2000-11-02</publish_date> <description>After an inadvertent trip through a Heisenberg Uncertainty Device, James Salway discovers the problem of being quantum.</description> <book id="bk110"> <author>O'Brien, Tim</author> <title>Microsoft .NET: The Programming Bible</title> <genre>Computer</genre> <price>36.95</price> <publish_date>2000-12-09</publish_date> <description>Microsoft's .NET initiative is explored in detail in this deep programmer's reference.</description> <book id="bk111"> <author>O'Brien, Tim</author> <title>MSXML3</title> <genre>Computer</genre> <price>49.95</price> <publish_date>2001-04-16</publish_date> <description>The Microsoft MSXML3 parser is covered in detail, with attention to XML DOM interfaces, XSLT, and schema validation. Includes samples for VB, C++, and Java. Includes CD-ROM with source code and sample files.</description> </catalog>
```

Formato do Pedido SOAP enviado pelo browser

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header>
    <ns1:obtemXML xmlns:ns1="http://calculator.webservices.org/">
  </S:Header>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><?s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<s:Header>
<s:To>urn:uuid:12345678-1234-1234-1234-1234567890ab</s:To>
<s:Action>urn:calculator:doubleXMLResponse</s:Action>
<s:MessageID>urn:uuid:12345678-1234-1234-1234-1234567890ab</s:MessageID>
<s:ReplyTo>
<s:Address>urn:uuid:12345678-1234-1234-1234-1234567890ab</s:Address>
</s:ReplyTo>
<s:SoapAction>urn:calculator:doubleXMLResponse</s:SoapAction>
<?xml version="1.0"?>
<catalog>
<book id="bk101">
<author>Gambardella, Matthew</author>
<title>XML Developer's Guide</title>
<category>Computer / General</category>

```



SOAP – JAVA

Exemplo nº2 – Devolver o ficheiro Books.xml para leitura

Testes – Browser (Opção “Test Web Service” do Netbeans)

The screenshot shows the 'Method invocation trace' tab in the Netbeans IDE. The title bar says 'localhost:8080/CalculatorWSApplication/obtemXMLTester'. The main area displays the 'obtemXML Method invocation' section.

Method parameter(s)

Type	Value
------	-------

Method returned

```
java.lang.String : <?xml version="1.0"?><catalog><book id="bk101"><author>Gambardella, Matthew</author><title>XML Developer's Guide</title><genre>Computer</genre><price>44.95</price><publish_date>2000-10-01</publish_date><description>An in-depth look at creating applications with XML.</description><book id="bk102"><author>Ralls, Kim</author><title>Midnight Rain</title><genre>Fantasy</genre><price>5.95</price><publish_date>2000-12-16</publish_date><description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description><book id="bk103"><author>Corets, Eva</author><title>Maeve Ascendant</title><genre>Fantasy</genre><price>5.95</price><publish_date>2000-11-17</publish_date><description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description><book id="bk104"><author>Corets, Eva</author><title>Oberon's Legacy</title><genre>Fantasy</genre><price>5.95</price><publish_date>2001-03-10</publish_date><description>In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Maeve Ascendant.</description><book id="bk105"><author>Corets, Eva</author><title>The Sundered Grail</title><genre>Fantasy</genre><price>5.95</price><publish_date>2001-09-10</publish_date><description>The two daughters of Maeve, half-sisters, battle one another for control of England. Sequel to Oberon's Legacy.</description><book id="bk106"><author>Randall, Cynthia</author><title>Lover Birds</title><genre>Romance</genre><price>4.95</price><publish_date>2000-09-02</publish_date><description>When Carla meets Paul at an ornithology conference, tempers fly as feathers get ruffled.</description><book id="bk107"><author>Thurman, Paula</author><title>Splish Splash</title><genre>Romance</genre><price>4.95</price><publish_date>2000-11-02</publish_date><description>A deep sea diver finds love twenty thousand leagues beneath the sea.</description><book id="bk108"><author>Kaor, Stefan</author><title>Creepy Crawlies</title><genre>Horror</genre><price>4.95</price><publish_date>2000-12-06</publish_date><description>An anthology of horror stories about reechtes, centipedes, scorpions and other insects.</description><book id="bk109"><author>Kress, Peter</author><title>Paradox Lost</title><genre>Science Fiction</genre><price>6.95</price><publish_date>2000-11-02</publish_date><description>After an inadvertant trip through a Heisenberg Uncertainty Device, James Salway discovers the problems of being quantum.</description><book id="bk110"><author>O'Brien, Tim</author><title>Microsoft .NET: The Programming Bible</title><genre>Computer</genre><price>36.95</price><publish_date>2000-12-09</publish_date><description>Microsoft's .NET initiative is explored in detail in this deep programmer's reference.</description><book id="bk111"><author>O'Brien, Tim</author><title>MSXML3: A Comprehensive Guide</title><genre>Computer</genre><price>36.95</price><publish_date>2000-12-01</publish_date><description>The Microsoft MSXML3 parser is covered in detail, with attention to XML DOM interfaces, XSLT processing, SAX and more.</description><book id="bk112"><author>Galos, Mike</author><title>Visual Studio 7: A Comprehensive Guide</title><genre>Computer</genre><price>49.95</price><publish_date>2001-04-16</publish_date><description>Microsoft Visual Studio 7 is explored in depth, looking at how Visual Basic, Visual C++, C#, and ASP are integrated into a comprehensive development environment.</description></catalog>
```

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/>
<S:Body>
    <ns2:obtemXML xmlns:ns2="http://calculator.webservices.com.br">
        </S:Body>
    </S:Envelope>
```

Formato da resposta SOAP dada pelo servidor

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Header/>
<S:Body>
    <ns2:obtemXMLResponse xmlns:ns2="http://calculator.webservices.com.br">
        <return><?xml version="1.0"?>
<catalog>
<book id="bk101">
<author>Gambardella, Matthew</author>
<title>XML Developer's Guide</title>
<genre>Computer</genre>
<price>44.95</price>
</book>
</catalog>
```



SOAP – JAVA

Exemplo nº2 – Devolver o ficheiro Books.xml para leitura

Testes – SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane, titled 'Request 1', displays the XML envelope sent to the service. The right pane, titled 'Response', shows the XML response received from the service, which is the 'Books.xml' file.

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header>
    <cal:obtemXML xmlns:cal="http://calculator.ws.org/cal">
      <return>i
    </cal:obtemXML>
  </soapenv:Header>
  <soapenv:Body>
    <cal:obtemXML>
      <return><?xml version="1.0"?>
<catalog>
  <book id="bk101">
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>A in-depth look at creating applications with XML.</description>
  </book>
  <book id="bk102">
    <author>Hall, Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-12-16</publish_date>
    <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
  </book>
  <book id="bk103">
    <author>Corets, Eva</author>
    <title>Maeve Ascendant</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-11-17</publish_date>
    <description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description>
  </book>
  <book id="bk104">
    <author>Corets, Eva</author>
    <title>Oberon's Legacy</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2001-03-10</publish_date>
    <description>In post-apocalypse England, the mysterious agents known only as Oberon have to create a new life</description>
  </book>
</catalog>
</?xml>
</return>
</cal:obtemXML>
</Body>
</Envelope>
```

Pedido SOAP Enviado



SOAP – JAVA

Exemplo nº2 – Devolver o ficheiro Books.xml para leitura

Testes – SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane displays the XML request sent to the service endpoint `http://localhost:8080/CalculatorWSApplication/obtemXML`. The right pane displays the XML response received from the service, which is the `Books.xml` file. The response XML is highlighted with a red box.

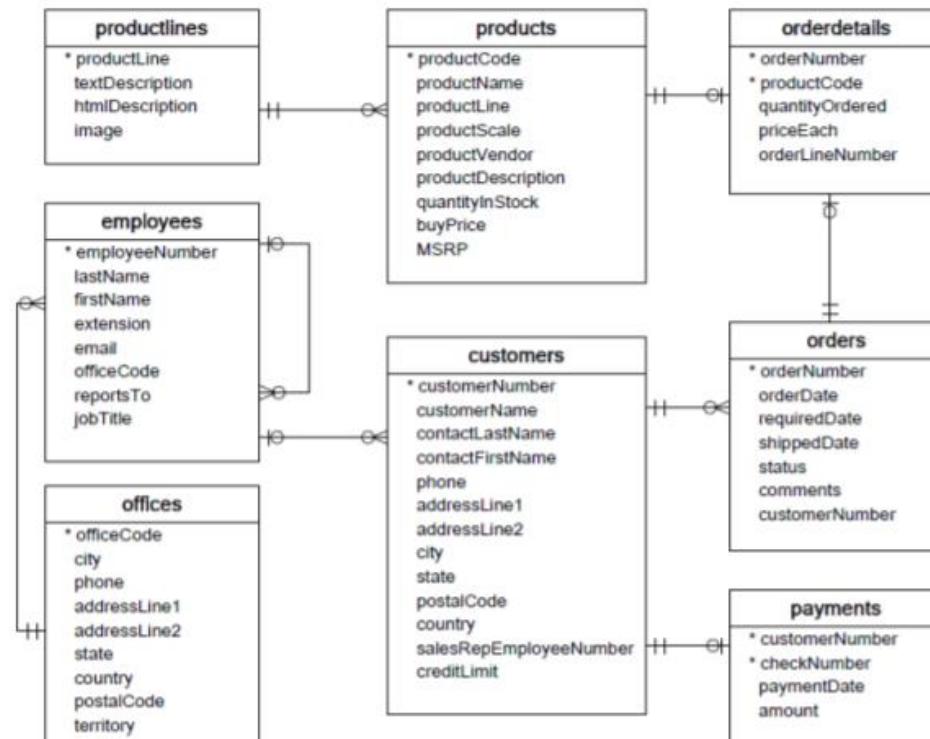
```
<?xml version="1.0"?>
<catalog>
    <book id="bk101">
        <author>Gambardella, Matthew</author>
        <title>XML Developer's Guide</title>
        <genre>Computer</genre>
        <price>44.95</price>
        <publish_date>2000-10-01</publish_date>
        <description>An in-depth look at creating applications with XML.</description>
    </book>
    <book id="bk102">
        <author>Hall, Kim</author>
        <title>Midnight Rain</title>
        <genre>Fantasy</genre>
        <price>5.95</price>
        <publish_date>2000-12-16</publish_date>
        <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
    </book>
    <book id="bk103">
        <author>Corets, Eva</author>
        <title>Heire Ascendant</title>
        <genre>Fantasy</genre>
        <price>5.95</price>
        <publish_date>2000-11-17</publish_date>
        <description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description>
    </book>
    <book id="bk104">
        <author>Cox, Eva</author>
        <title>Oberon's Legacy</title>
        <genre>Fantasy</genre>
        <price>5.95</price>
        <publish_date>2001-03-10</publish_date>
        <description>In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life.
```

Resposta SOAP

SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

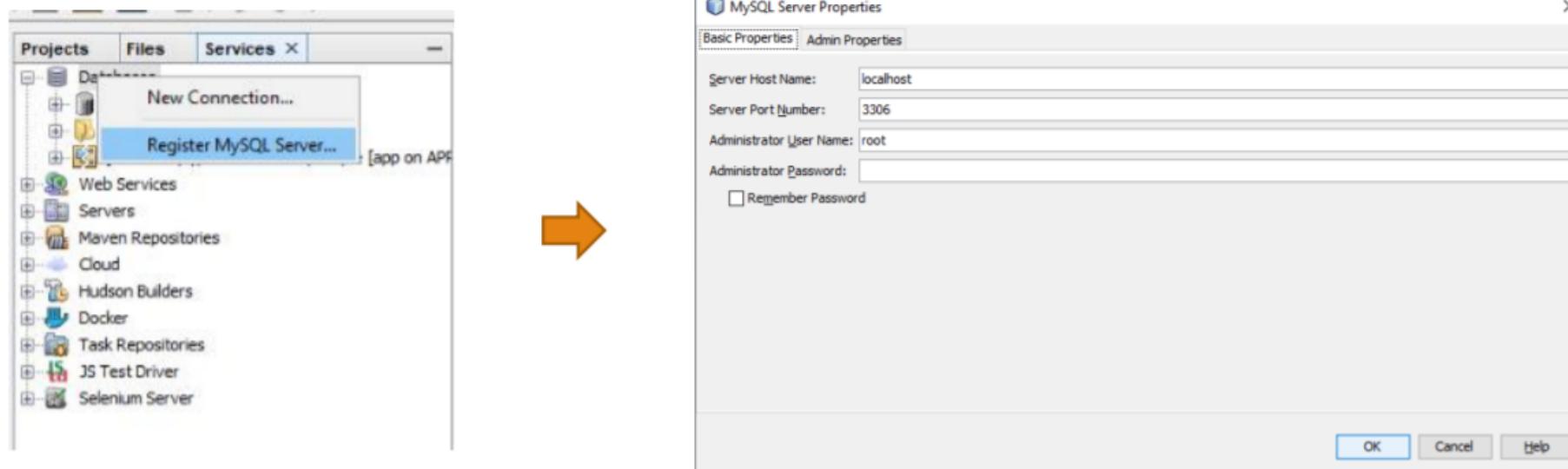
Os registos utilizados neste exemplo encontram-se disponíveis na base de dados de testes disponível em <http://www.mysqltutorial.org/mysql-sample-database.aspx>



SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

Inicialmente, é necessário criar a ligação à base de dados através do conector JDBC.



https://www.youtube.com/watch?v=ET6BIRZY_Jw



SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

Após criar a ligação, é necessário criar um projeto do tipo “Java Web Application”. Nesse novo projeto, deve ser criada uma classe “session bean”, na qual ficará o método que acede à base de dados para obter informação. Esse método, terá o seguinte código:

```
@Stateless
public class NewSessionBean {

    public Employees obtemEmployees() {
        Employees employees = new Employees();

        //String driver_JDBC = "com.mysql.cj.jdbc.Driver";
        String url_JDBC = "jdbc:mysql://localhost:3306/classicmodels";
        String utilizador = "root";
        String pass = "";
        try{
            Class.forName("com.mysql.jdbc.Driver").newInstance();
            Connection conn = DriverManager.getConnection(url_JDBC, utilizador, pass);
            Statement stmt = conn.createStatement();
            ResultSet result = stmt.executeQuery("SELECT * FROM employees");
            while(result.next())
                employees.getEmployees().add(Employee.criarEmployee(result.getInt(1), result.getString(3), result.getString(2)));
            conn.close();
        } catch(Exception e){
            System.out.println("Ocorreu um erro" + e.getMessage());
            employees.getEmployees().add(Employee.criarEmployee(1, "OCORREU UM ERRO - ", e.getMessage()));
        }
        return employees;
    }
}
```



SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

São necessárias também as seguintes classes:

```
/*
@XmlRootElement(name="employee")
public class Employee implements Serializable{
    private int identificador;
    private String primeironome;
    private String ultimonome;

    public Employee() {}

    public Employee(int identificador, String primeironome, String ultimonome) {
        this.identificador = identificador;
        this.primeironome = primeironome;
        this.ultronome = ultimonome;
    }

    public static Employee criarEmployee(int identificador, String primeironome, String ultimonome) {
        return new Employee(identificador, primeironome, ultimonome);
    }

    public int getIdentificador() {
        return identificador;
    }

    public void setIdentificador(int identificador) {
        this.identificador = identificador;
    }

    public String getPrimeironome() {
        return primeironome;
    }

    public void setPrimeironome(String primeironome) {
        this.primeironome = primeironome;
    }

    public String getUltimonome() {
        return ultimonome;
    }

    public void setUltimonome(String ultimonome) {
        this.ultronome = ultimonome;
    }
}
```



SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

e:

```
@XmlRootElement(name = "employees")
public class Employees {
    private List<Employee> employees = new ArrayList<>();

    public Employees() {
    }

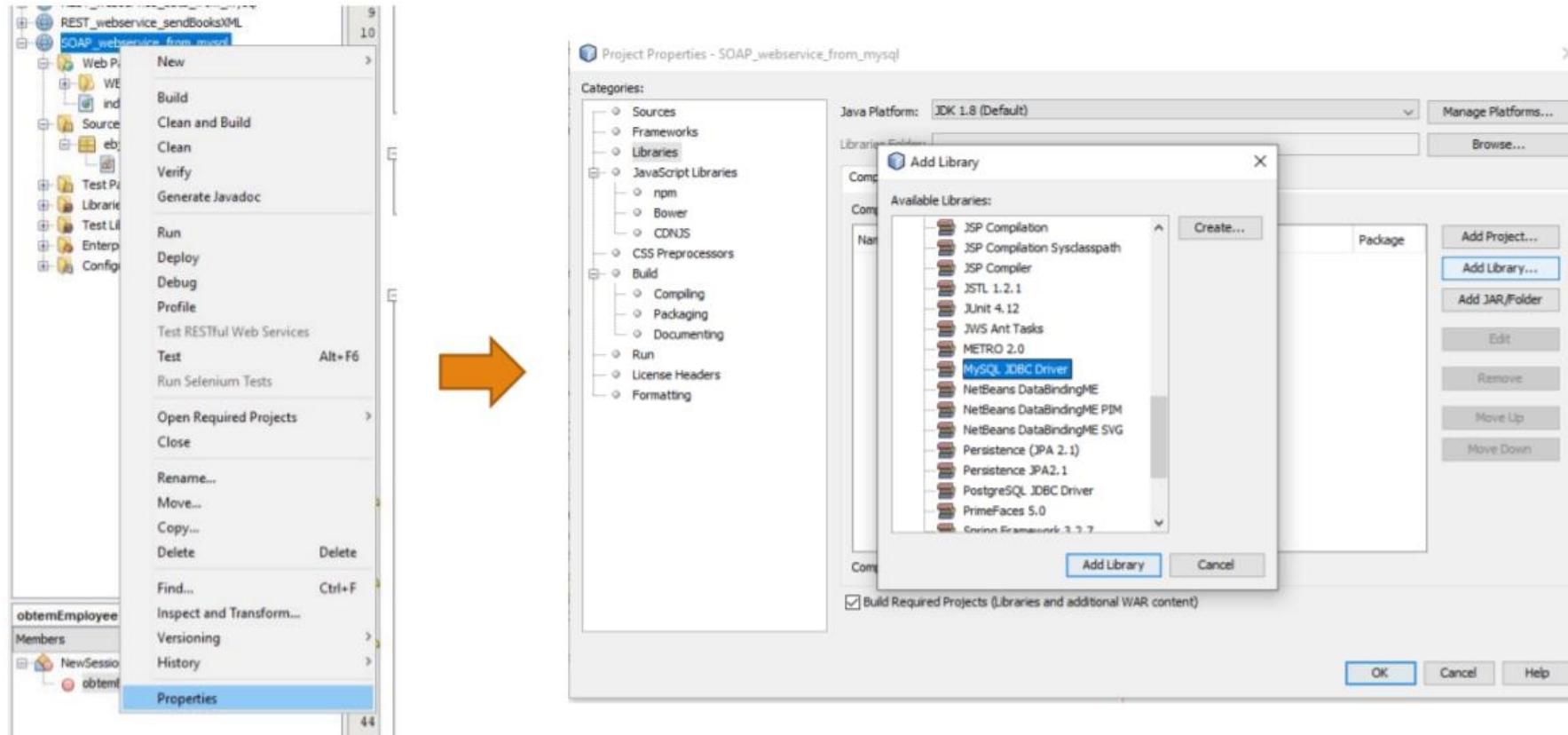
    public List<Employee> getEmployees() {
        return employees;
    }

    public void setEmployees(List<Employee> employees) {
        this.employees = employees;
    }
}
```

SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

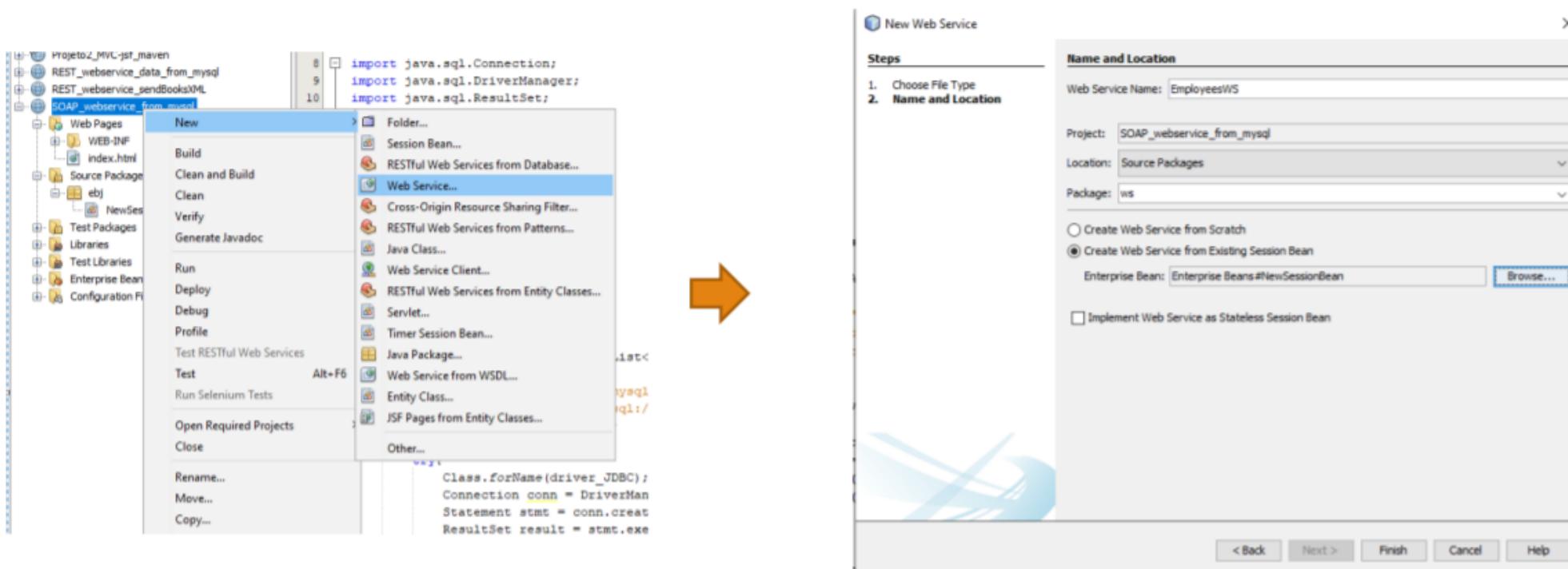
A libraria do Driver Mysql deve ser associada ao projeto



SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

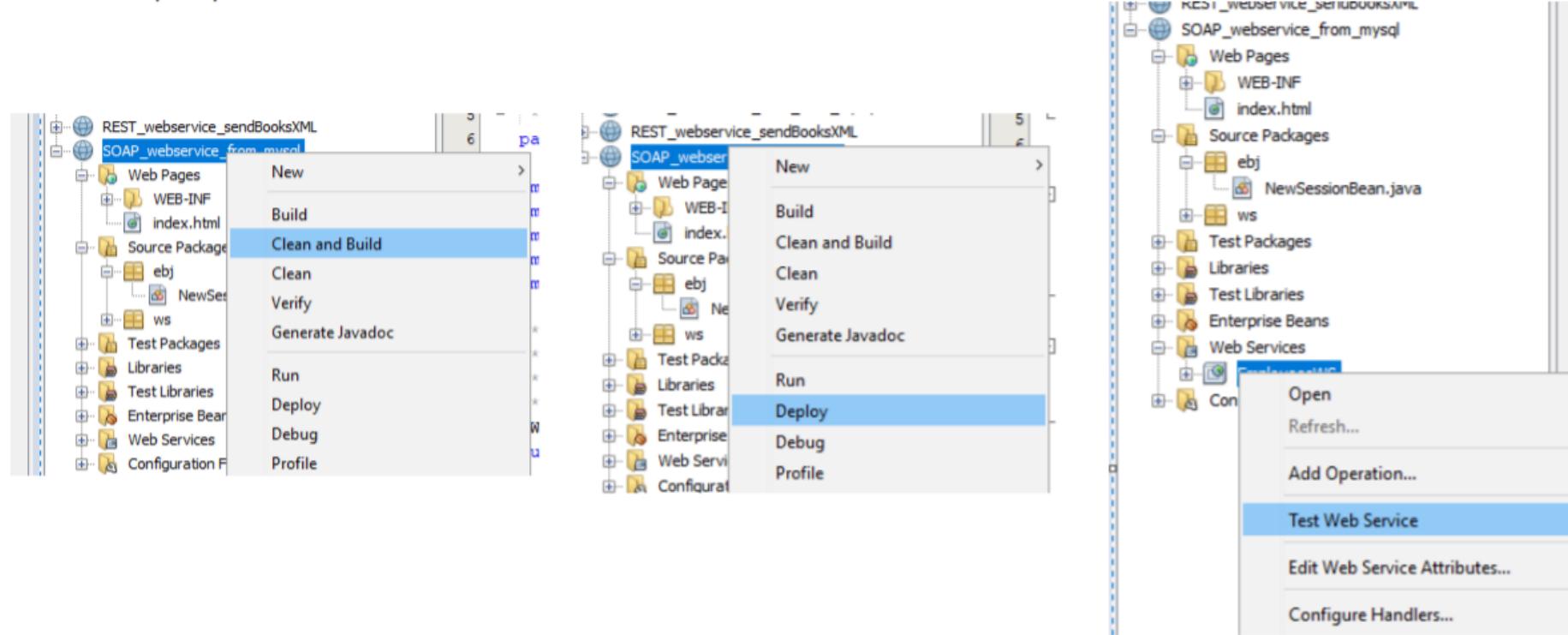
Após criada a ligação, session bean e adicionada a libraria, é necessário criar um novo web service para usar a session bean.



SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

Agora, é necessário Construir o projeto através do “Build”, e dar Deploy ao mesmo, para que possa ser testado.



SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

Testes – Browser (Opção “Test Web Service” do Netbeans)

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

Methods :

```
public abstract ws.Employees ws.EmployeesWS.obtemEmployees()
obtemEmployees 0
```

Method invocation trace

obtemEmployees Method invocation

Method parameter(s)

Type	Value
------	-------

Method returned

```
ws.Employees : "ws.Employees@2d5fla1f"
```

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope">
<S:Header>
<S:Body>
<ns2:obtemEmployees xmlns:ns2="http://ws/">
</S:Body>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope">
<S:Header>
<S:Body>
<ns2:obtemEmployeesResponse xmlns:ns2="http://ws/">
<return>
<employees>
<employee>
<identificador>1002</identificador>
<primeironome>Diane</primeironome>
<ultimonome>Murphy</ultimonome>
</employee>
<employee>
<identificador>1056</identificador>
<primeironome>Mary</primeironome>
<ultimonome>Patterson</ultimonome>
</employee>
<employee>
<identificador>1076</identificador>
<primeironome>Jeff</primeironome>
<ultimonome>Firrelli</ultimonome>
</employee>
<employee>
<identificador>1088</identificador>
<primeironome>William</primeironome>
<ultimonome>Patterson</ultimonome>
</employee>
<employee>
<identificador>1102</identificador>
<primeironome>Lorraine</primeironome>
<ultimonome>Harris</ultimonome>
</employee>
<employees>
</employees>
</return>
</ns2:obtemEmployeesResponse>
</S:Body>
</S:Envelope>
```



SOAP – JAVA

Exemplo nº3 – Devolver dados da base de dados

Testes – SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane, titled 'Request 1', displays the XML of a SOAP message sent to the endpoint `http://localhost:8080/SOAP_webservice_from_mysql/EmployeesWS`. The message contains a single operation `<ns1:obtemEmployees/>`. The right pane, titled 'Raw XML', shows the XML response received from the server. The response is a multi-level structure starting with `<ns1:obtemEmployeesResponse>`, which contains multiple `<employees>` blocks, each representing an employee with attributes like `identificador`, `primeironome`, and `ultimonome`.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope">
<soapenv:Header/>
<soapenv:Body>
<ns1:obtemEmployees/>
</soapenv:Body>
</soapenv:Envelope>

<?xml version="1.0" encoding="UTF-8"?>
<ns1:obtemEmployeesResponse xmlns:ns1="http://ws/">
<return>
<employees>
<employee>
<identificador>1002</identificador>
<primeironome>Diane</primeironome>
<ultimonome>Murphy</ultimonome>
</employee>
<employee>
<identificador>1056</identificador>
<primeironome>Mary</primeironome>
<ultimonome>Patterson</ultimonome>
</employee>
<employee>
<identificador>1076</identificador>
<primeironome>Jeff</primeironome>
<ultimonome>Firrelli</ultimonome>
</employee>
<employee>
<identificador>1088</identificador>
<primeironome>William</primeironome>
<ultimonome>Patterson</ultimonome>
</employee>
<employee>
<identificador>1102</identificador>
<primeironome>Gerard</primeironome>
<ultimonome>Bondur</ultimonome>
</employee>
<employee>
<identificador>1143</identificador>
<primeironome>Anthony</primeironome>
<ultimonome>Bow</ultimonome>
</employee>
<employee>
<identificador>1165</identificador>
<primeironome>Leslie</primeironome>
<ultimonome>Jennings</ultimonome>
</employee>
<employee>
<identificador>1166</identificador>
<primeironome>Leslie</primeironome>
<ultimonome>Leslie</ultimonome>
</employee>
</employees>
</ns1:obtemEmployeesResponse>
</?xml>
```

Auth Headers (0) Attachments (0) WSS-A WSS-RM JMS Headers JMS Property (0)
Headers (0) Attachments (0) SSL Info WSS (0) JMS (0)
response time: 18ms (3076 bytes)



SOAP – PHP

Exemplo nº1: Exemplo dado pelo sitepoint.com

Código Servidor (envio da informação + criação automática do ficheiro wsdl)

```
SendBooks.php X  ReceiveBooks.php
SendBooks.php > ...
1  <?php
2  require_once "nusoap.php";
3  function getProd($category) {
4      if ($category == "books") {
5          return join(", ", array(
6              "The WordPress Anthology",
7              "PHP Master: Write Cutting Edge Code",
8              "Build Your Own Website the Right Way"));
9      }
10     else {
11         return "No products listed under that category";
12     }
13 }
14
15 $server = new soap_server();
16 $server->configureWSDL('AddService', 'ObtemBooksString');
17 $server->wsdl->schemaTargetNamespace = 'ObtemBooksString';
18 $server->register("getProd",
19 |     array(),
20 |     array('return' => 'xsd:string')
21 );
22 $server->service(file_get_contents('php://input'));
23
24
25 ?>
```



SOAP – PHP

Exemplo nº1: Exemplo dado pelo sitepoint.com

Código Cliente

```
SendBooks.php    ReceiveBooks.php X
ReceiveBooks.php > ...
1  <?php
2  require_once "nusoap.php";
3  $client = new nusoap_client("http://localhost/IS WebServices/SendBooks.php");
4
5  $error = $client->getError();
6  if ($error) {
7      echo "<h2>Constructor error</h2><pre>" . $error . "</pre>";
8  }
9
10 $result = $client->call("getProd", array("category" => "books"));
11
12 if ($client->fault) {
13     echo "<h2>Fault</h2><pre>";
14     print_r($result);
15     echo "</pre>";
16 }
17 else {
18     $error = $client->getError();
19     if ($error) {
20         echo "<h2>Error</h2><pre>" . $error . "</pre>";
21     }
22     else {
23         echo "<h2>Books</h2><pre>";
24         echo $result;
25         echo "</pre>";
26     }
27 }
28
29 ?>
```



SOAP – PHP

Exemplo nº1: Exemplo dado pelo sitepoint.com

Teste – Broswer (Cliente php)

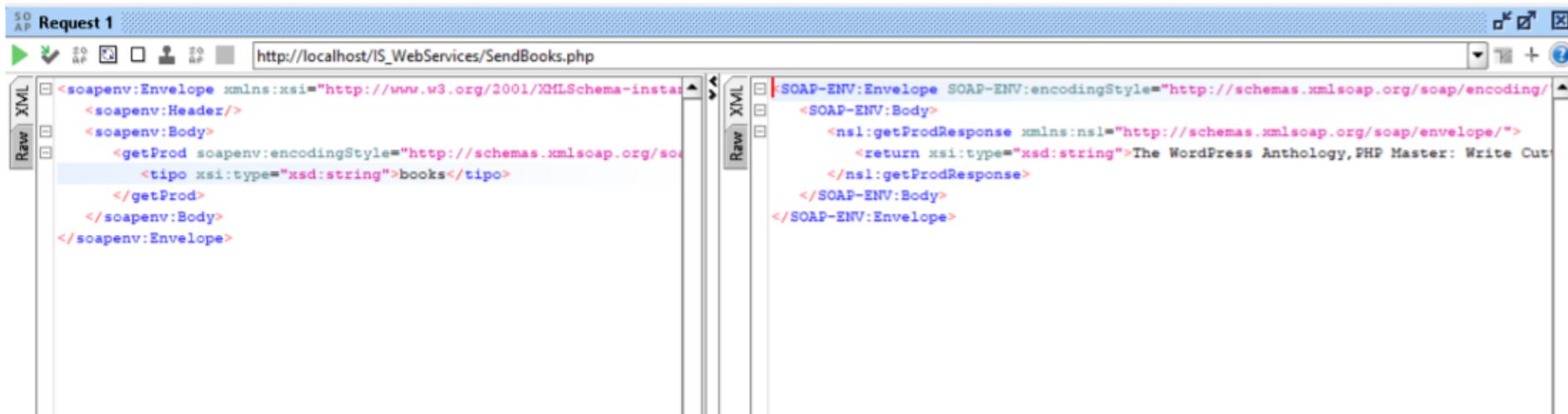
Books

The WordPress Anthology, PHP Master: Write Cutting Edge Code, Build Your Own Website the Right Way

SOAP – PHP

Exemplo nº1: Exemplo dado pelo sitepoint.com

Teste – SOAP UI



The screenshot shows the SOAP UI interface with two panes. The left pane displays the XML request sent to the service, and the right pane displays the XML response received from the service.

Request:

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns1="http://schemas.xmlsoap.org/soap/encoding/">
  <soapenv:Header/>
  <soapenv:Body>
    <getProd soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <tipo xsi:type="xsd:string">books</tipo>
    </getProd>
  </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <ns1:getProdResponse xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/">
      <return xsi:type="xsd:string">The WordPress Anthology, PHP Master: Write Cut
    </ns1:getProdResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```



SOAP – PHP

Exemplo nº2: Devolver o ficheiro Books.xml para leitura

Código Servidor – Baseado no exemplo 1

```
SendBooksXML.php > ...
1  <?php
2  require_once "nusoap.php";
3  function getProd($category) {
4      if ($category == "books") {
5          return file_get_contents("books.xml");
6      }
7      else {
8          return "No products listed under that category";
9      }
10 }
11
12 $server = new soap_server();
13 $server->configureWSDL('AddService', 'ObtemBooksString');
14 $server->wsdl->schemaTargetNamespace = 'ObtemBooksString';
15 $server->register("getProd",
16     array('tipo' => 'xsd:string'),
17     array('return' => 'xsd:xml')
18 );
19 $server->service(file_get_contents('php://input'));
20 ?>
```



SOAP – PHP

Exemplo nº2: Devolver o ficheiro Books.xml para leitura

Teste – SOAP UI

The screenshot shows the SoapUI interface with two panes. The left pane, titled 'Request 1', displays a SOAP message in XML format:

```
<?xml version="1.0" encoding="ISO-8859-1"?><SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
<SOAP-ENV:Header/>
<SOAP-ENV:Body>
<getProd soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
<tipo xsi:type="xsd:string">books</tipo>
</getProd>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The right pane, titled 'Response', displays the XML response received from the service:

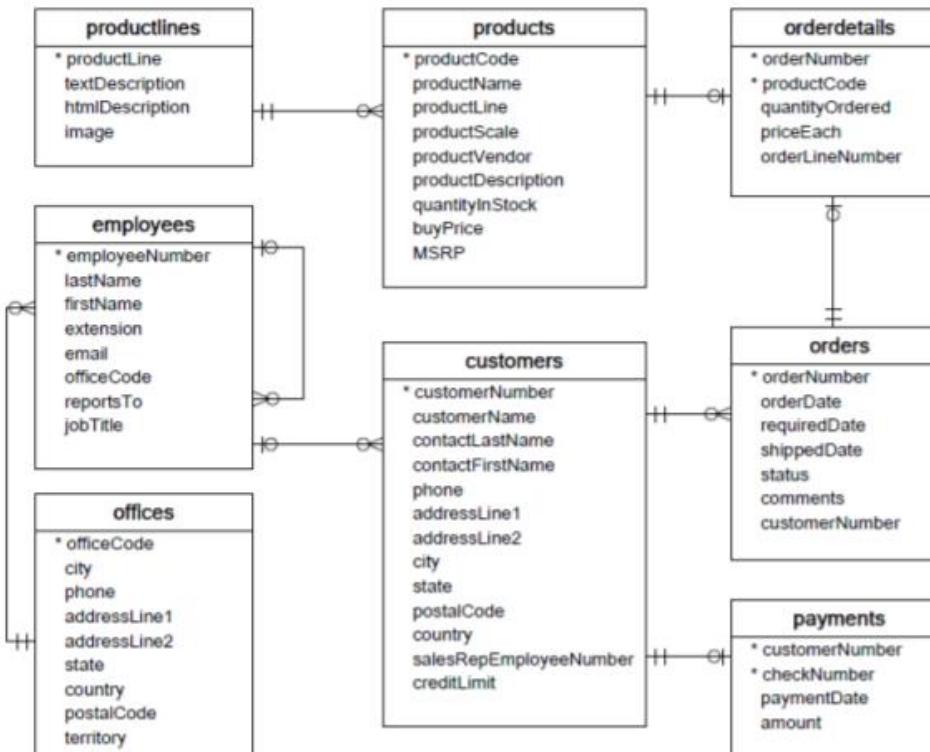
```
<?xml version="1.0" encoding="ISO-8859-1"?><SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
<catalog>
<book id="bk101">
<author>Gamberdella, Matthew</author>
<title>XML Developer's Guide</title>
<genre>Computer</genre>
<price>44.95</price>
<publish_date>2000-10-01</publish_date>
<description>An in-depth look at creating applications with XML.</description>
</book>
<book id="bk102">
<author>Wall, Kim</author>
<title>Midnight Rain</title>
<genre>Fantasy</genre>
<price>5.95</price>
<publish_date>1000-12-16</publish_date>
<description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
</book>
<book id="bk103">
<author>Corets, Eva</author>
<title>Maeve Ascendant</title>
<genre>Fantasy</genre>
<price>5.95</price>
<publish_date>2000-11-17</publish_date>
<description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description>
</book>
<book id="bk104">
<author>Corets, Eva</author>
<title>Oberon's Legacy</title>
<genre>Fantasy</genre>
<price>5.95</price>
<publish_date>2001-09-10</publish_date>
<description>In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Maeve Ascendant.</description>
</book>
</catalog>
</SOAP-ENV:Envelope>
```

At the bottom of the interface, there are tabs for 'Auth', 'Headers (0)', 'Attachments (0)', 'WS-A', 'WS-RM', 'JMS Headers', 'JMS Property (0)', 'Headers (10)', 'Attachments (0)', 'SSL Info', 'WSS (0)', 'JMS (0)', and status indicators for 'response time: 6ms (5080 bytes)' and '1:1'.

SOAP – PHP

Exemplo nº3: Devolver registos da base de dados em XML

Os registos utilizados neste exemplo encontram-se disponíveis na base de dados de testes disponível em <http://www.mysqltutorial.org/mysql-sample-database.aspx>





SOAP – PHP

Exemplo nº3: Devolver registos da base de dados em XML

Código Servidor – Parte 1

```
<?php
require_once "nusoap.php";
function getBooks() {
    /* máximo de registos */
    $number_of_posts = 10;
    /* número de registo de inicio a listar */
    $inicio = 1 * $number_of_posts;

    /* ligar à base de dados */
    $link = mysqli_connect('localhost', 'root', '') or die('Cannot connect to the DB');
    mysqli_select_db($link, 'classicmodels') or die('Cannot select the DB');

    /* obtem dados */
    $query = "SELECT customerNumber, customerName, phone, addressLine1, city, country FROM customers LIMIT $inicio, $number_of_posts";
    $result = mysqli_query($link, $query) or die('Errant query: ' . $query);

    $arr = array();
    if(mysqli_num_rows($result)) {
        while($linha = mysqli_fetch_assoc($result)) {
            $arr[] = array('customers'=>$linha);
        }
    }
}
```



SOAP – PHP

Exemplo nº3: Devolver registos da base de dados em XML

Código Servidor – Parte 2

```
$stringXML = "";

header('Content-type: text/xml');
$stringXML .= '<customerss>; // tag inicial do documento
foreach($arr as $index => $cliente) { // percorre array obtido da base de dados
    if(is_array($cliente)) {
        foreach($cliente as $key => $value) { // cada $cliente é um registo da base de dados representado num array de 1 elemento | obtem esse elemento para $value
            $stringXML .= '<'.$key.'>; // inicia tag
            if(is_array($value)) {
                foreach($value as $tag => $val) { // percorre todos os elementos de cada registo obtido da base de dados
                    $stringXML .= '<'.$tag.'>'.htmlentities($val).'</'.$tag.'>; // echo a cada elemento
                }
            }
            $stringXML .= '</'.$key.'>; // termina a tag
        }
    }
}
$stringXML .= '</customerss>;

return $stringXML;
}

$server = new soap_server();
$server->configureWSdl('AddService', 'ObtemBooksString');
$server->wsdl->schemaTargetNamespace = 'ObtemBooksString';
$server->register("getBooks", array(), array('return' => 'xsd:string'));
$server->service(file_get_contents('php://input'));
?>
```



SOAP – PHP

Exemplo nº3: Devolver registos da base de dados em XML

Teste – SOAP UI

The screenshot shows the SoapUI application interface. On the left, there is a 'Request 1' panel with a green play button and a 'Raw XML' tab. The URL is http://localhost/IS_WebServices/SOAP_SendDatabaseData.php. The XML content is a SOAP envelope with a 'getBooks' body element. On the right, there is a 'Response 1' panel with a blue plus sign icon and a 'Raw XML' tab. The XML content is a SOAP response containing multiple customer records, each with a unique customer number, name, phone number, address, and city.

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://schemas.xmlsoap.org/soap/types/">
  <soapenv:Header/>
  <soapenv:Body>
    <getBooks soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
  </soapenv:Body>
</soapenv:Envelope>

<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Header/>
<ns1:getBooksResponse xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/">
<return xsi:type="xsd:string">
<customers>
  <customers>
    <customerNumber>141</customerNumber>
    <customerName>Euro+ Shopping Channel</customerName>
    <phone>(91) 555 94 44</phone>
    <addressLine1>C/ Moralsarzal, 86</addressLine1>
    <city>Madrid</city>
    <country>Spain</country>
  </customers>
  <customers>
    <customerNumber>144</customerNumber>
    <customerName>Volvo Model Replicas, Co</customerName>
    <phone>(921) 12 3555</phone>
    <addressLine1/>
    <city/>
    <country>Sweden</country>
  </customers>
  <customers>
    <customerNumber>145</customerNumber>
    <customerName>Danish Wholesale Imports</customerName>
    <phone>31 12 3555</phone>
    <addressLine1/>
    <city>Kopenhagen</city>
    <country>Denmark</country>
  </customers>
  <customers>
    <customerNumber>146</customerNumber>
    <customerName>Saveley Lamp, Henriot, Co.</customerName>
    <phone>78.32.5555</phone>
    <addressLine1>2, rue du Commerce</addressLine1>
    <city>Lyon</city>
    <country>France</country>
  </customers>
  <customers>
    <customerNumber>148</customerNumber>
    <customerName>Dragon Souvenirs, Ltd.</customerName>
    <phone>+65 221 7555</phone>
    <addressLine1>Raffles, Bldg. 1</addressLine1>
  </customers>
</customers>
</return>
</ns1:getBooksResponse>
</SOAP-ENV:Envelope>
```



REST – JAVA

Exemplo nº1: Devolver o ficheiro Books.xml para leitura

Código – Parte 1 (Utiliza o DOM Parser)

```
@Path("obtemXML")
public class ObtemXML {

    @Context
    private UriInfo context;

    public ObtemXML() {}

    @GET
    @Produces(MediaType.APPLICATION_XML)
    public String getXml() {
        try {
            File xmlFile = new File("D:\\IPVC\\5\\S. - Integração de Sistemas\\Trabalhos Práticos\\TPI - Webservices SOAP e REST\\JAVA\\REST_webservice\\books.xml");
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            Document doc = builder.parse(xmlFile);

            String xml = "";
            xml += parseWholeXML(doc.getDocumentElement());

            return xml;
        } catch (SAXException ex) {
            return "ERRO: " + ex.getMessage();
        } catch (IOException ex) {
            return "ERRO: " + ex.getMessage();
        } catch (ParserConfigurationException ex) {
            return "ERRO: " + ex.getMessage();
        }
    }
}
```

<https://netbeans.org/kb/docs/websvc/rest.html>

<https://www.mkyong.com/java/how-to-read-xml-file-in-java-dom-parser/>



REST – JAVA

Exemplo nº1: Devolver o ficheiro Books.xml para leitura

Código – Parte 2

```
private static String parseWholeXML(Node startingNode){  
    NodeList childNodes = startingNode.getChildNodes();  
    String s = "";  
    //inicializa tag  
    s += "<" + startingNode.getNodeName() + ">\n";  
  
    // percorre todos os filhos da tag dada  
    for(int i=0; i<childNodes.getLength(); i++){  
        Node childNode = childNodes.item(i);  
        if(childNode.getNodeType() == Node.ELEMENT_NODE){  
            // se possuir tag dentro da tag atual, executa recursivamente parseWholeXML() com a nova tag encontrada  
            s += parseWholeXML(childNode);  
        }else{  
            if(!childNode.getTextContent().trim().isEmpty()){  
                //adiciona texto que está dentro da tag  
                s += childNode.getTextContent() + "\n";  
            }  
        }  
    }  
    // finaliza tag  
    s += "</>" + startingNode.getNodeName() + ">\n";  
  
    return s;  
}
```



REST – JAVA

Exemplo nº1: Devolver o ficheiro Books.xml para leitura

Teste – Opção “Test RESTful Web Services” do Netbeans

The screenshot shows the Netbeans IDE interface. On the left, the Project Explorer displays a Maven project named 'Projeto2_MVC-jsf_maven'. A context menu is open over the 'REST' node, with the 'Test RESTful Web Services' option highlighted. An orange arrow points from this menu to the right-hand 'Test RESTful Web Services' dialog. The dialog shows the URL `http://localhost:8080/REST_webservice/webresources/application.wadl`. In the tree view, under 'REST_webservice > obtemXML > getFicheiroBooks.xml', the method 'getFicheiroBooks.xml' is selected. The 'Choose method to test:' dropdown is set to 'GET(application/octet-stream)'. Below the tree, the status is 'Status: 200 (OK)' and the response content is displayed in a tabular view:

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
  <book id="bk101">
    <author>Manoel, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications with XML.</description>
  </book>
  <book id="bk102">
    <author>Rafe Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-12-16</publish_date>
    <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
  </book>
  <book id="bk103">
    <author>Corets, Eva</author>
    <title>Maeve Ascendant</title>
    <genre>Fantasy</genre>
```



REST – JAVA

Exemplo nº1: Devolver o ficheiro Books.xml para leitura

Teste – POSTMAN

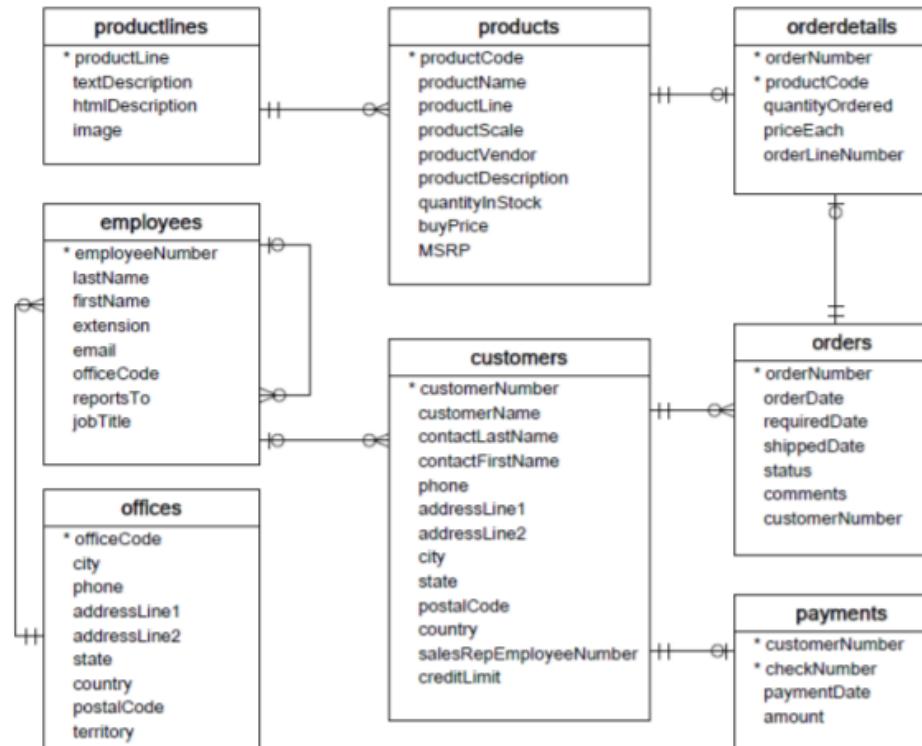
The screenshot shows the POSTMAN application interface. On the left, there's a sidebar with a history of requests and a list of APIs. The main area shows an 'Untitled Request' with a GET method pointing to `localhost:8080/REST_webService/webresources/bookXML/getFicheiroBooks.xml`. The 'Params' tab is selected, showing a single parameter 'key' with value 'Value'. The 'Body' tab shows the XML response:

```
<?xml version="1.0"?>
<catalog>
  <book id="bk101">
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications with XML.</description>
  </book>
  <book id="bk102">
    <author>Hall, Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <price>9.95</price>
    <publish_date>2000-12-16</publish_date>
    <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
  </book>
  <book id="bk103">
    <author>Curtis, Eve</author>
    <title>Warrior Ascendant</title>
    <genre>Fantasy</genre>
    <price>9.95</price>
    <publish_date>2000-11-17</publish_date>
    <description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description>
  </book>
  <book id="bk104">
    <author>Curtis, Eve</author>
    <title>Warrior's Heir</title>
```

REST – JAVA

Exemplo nº2: Acesso à base de dados e devolução de registo em XML

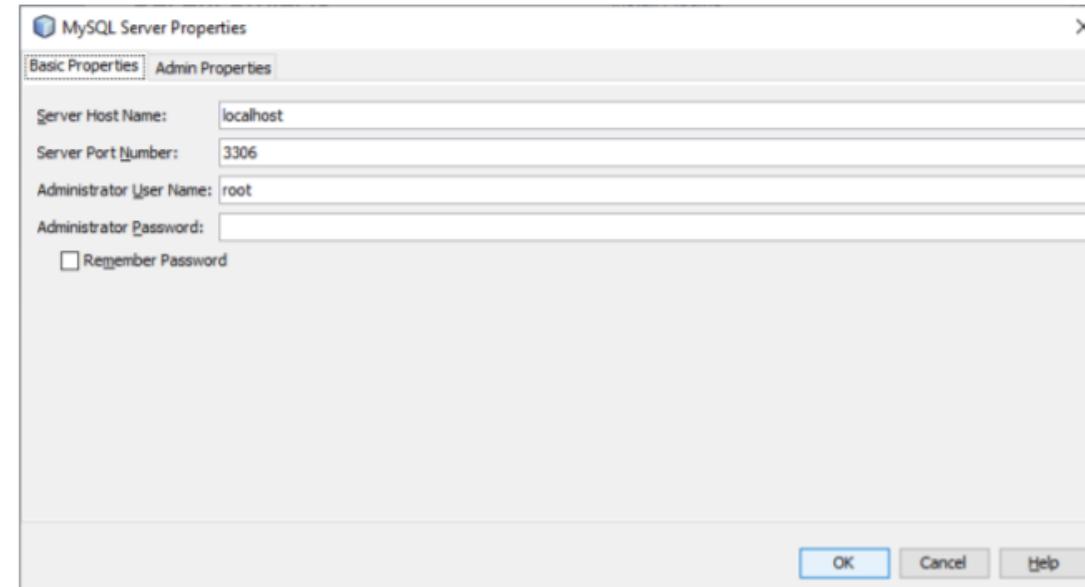
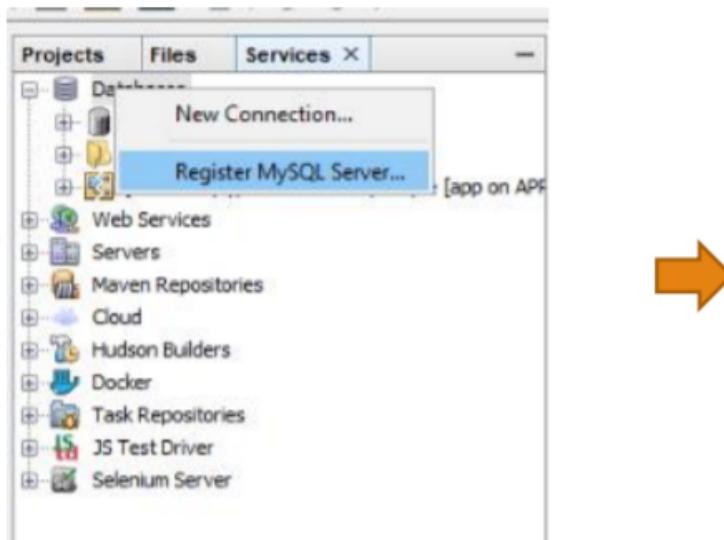
Os registos utilizados neste exemplo encontram-se disponíveis na base de dados de testes disponível em <http://www.mysqltutorial.org/mysql-sample-database.aspx>



REST – JAVA

Exemplo nº2: Acesso à base de dados e devolução de registos em XML

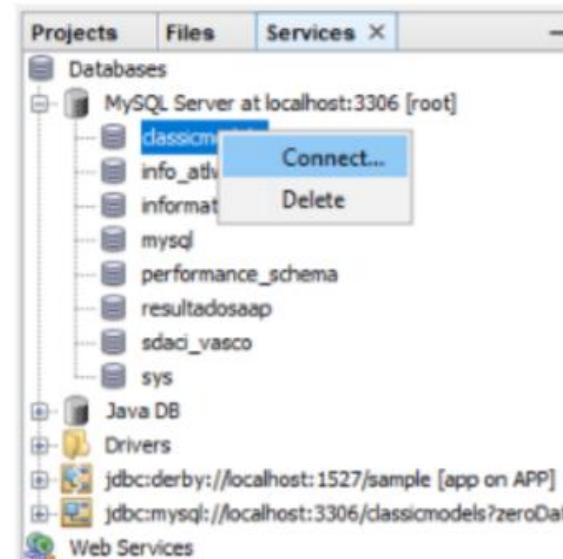
1 – Associar a base de dados mysql ao netbeans.



REST – JAVA

Exemplo nº2: Acesso à base de dados e devolução de registos em XML

2 – Adicionar a conexão à base de dados.

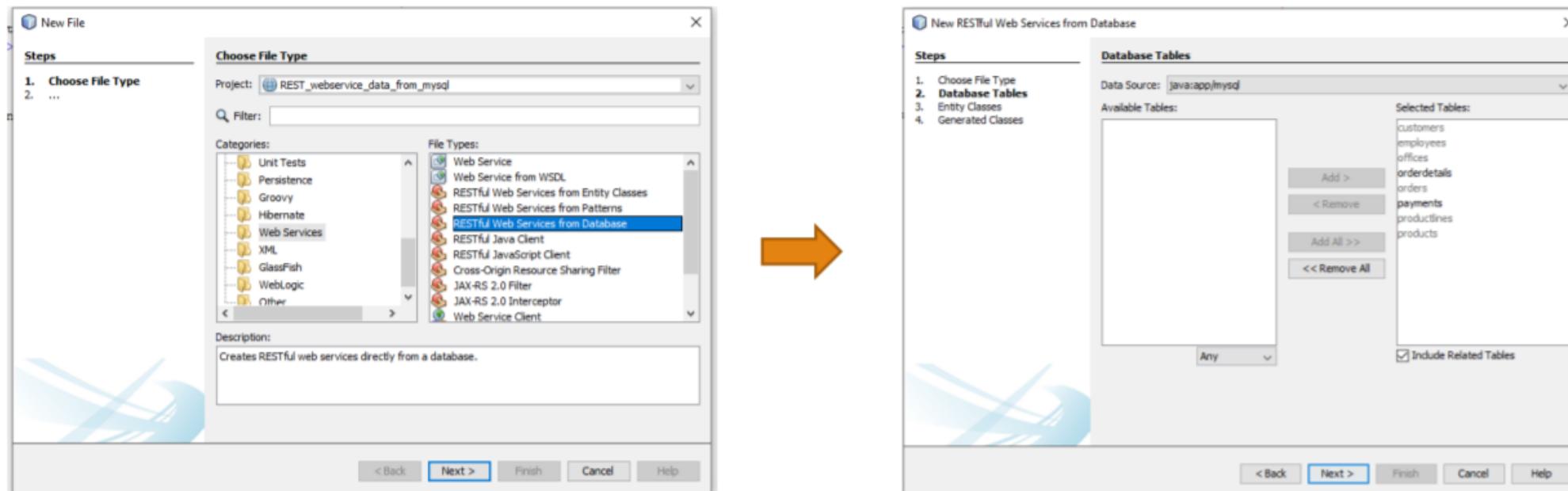




REST – JAVA

Exemplo nº2: Acesso à base de dados e devolução de registos em XML

3 – Criar uma Java Web > Web Application e gerar classes a partir da base de dados





REST – JAVA

Exemplo nº2: Acesso à base de dados e devolução de registo em XML

Exerto do código gerado, permite utilizar os web services com XML ou JSON

```
@GET  
@Path("{id}")  
@Produces({MediaType.APPLICATION_XML, MediaType.APPLICATION_JSON})  
public Customers find(@PathParam("id") Integer id) {  
    return super.find(id);  
}  
  
@GET  
@Override  
@Produces({MediaType.APPLICATION_XML, MediaType.APPLICATION_JSON})  
public List<Customers> findAll() {  
    return super.findAll();  
}  
  
@GET  
@Path("{from}/{to}")  
@Produces({MediaType.APPLICATION_XML, MediaType.APPLICATION_JSON})  
public List<Customers> findRange(@PathParam("from") Integer from, @PathParam("to") Integer to) {  
    return super.findRange(new int[]{from, to});  
}
```

REST – JAVA

Exemplo nº2: Acesso à base de dados e devolução de registo em XML

TESTE: Browser (Página gerada pelo netbeans)

The screenshot shows the NetBeans IDE interface. On the left, the Project Explorer displays several Java projects, including 'CalculatorWS_Client_Application', 'CalculatorWSApplication', 'Projeto2_Hibernate', 'Projeto2_JavaFX', 'Projeto2_MVC-jsf_maven', and 'REST_webservice_data_from_mysql'. The 'REST_webservice_data_from_mysql' project is selected, and its context menu is open, with 'Test RESTful Web Services' highlighted. An orange arrow points from this menu to the main window. The main window title is 'Test RESTful Web Services' with the URL 'http://localhost:8080/REST_webservice_data_from_mysql/webresources/application.wadl'. The central pane shows the WADL document structure under 'REST_webservice_data_from_mysql > classes.customers'. Below this, a 'Resource' section shows 'classes.customers' with the URL 'http://localhost:8080/REST_webservice_data_from_mysql/webresources/classes.customers'. A dropdown menu 'Choose method to test' is set to 'GET(application/xml)'. The right pane displays the response status 'Status: 200 (OK)' and the XML response content:

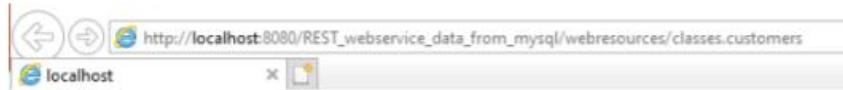
```
<?xml version="1.0" encoding="UTF-8"?>
<customers>
    <customers>
        <addressLine1>54, rue Royale</addressLine1>
        <city>Nantes</city>
        <contactFirstName>Carine</contactFirstName>
        <contactLastName>Schmitt</contactLastName>
        <country>France</country>
        <creditLimit>21000.00</creditLimit>
        <customerName>Atelier graphique</customerName>
        <customerNumber>103</customerNumber>
        <employeeNumber>40 32 2555</employeeNumber>
        <officeCode>4499</officeCode>
        <salesRepEmployeeNumber>
            <email>ghemande@classicmodelcars.com</email>
            <employeeNumber>1370</employeeNumber>
            <extension>2923</extension>
            <firstName>Gerard</firstName>
            <jobTitle>Sales Rep</jobTitle>
            <lastName>Hernandez</lastName>
            <officeCode>
                <addressLine1>43 Rue Joffrey D'abbans</addressLine1>
            
```



REST – JAVA

Exemplo nº2: Acesso à base de dados e devolução de registo em XML

TESTE: Browser (Link direto)



```
<?xml version="1.0" encoding="UTF-8" standalone="true"?>
- <customers>
  - <customers>
    <addressLine1>54, rue Royale</addressLine1>
    <city>Nantes</city>
    <contactFirstName>Carine </contactFirstName>
    <contactLastName>Schmitt</contactLastName>
    <country>France</country>
    <creditLimit>21000.00</creditLimit>
    <customerName>Atelier graphique</customerName>
    <customerNumber>103</customerNumber>
    <phone>40.32.2555</phone>
    <postalCode>44000</postalCode>
    - <salesRepEmployeeNumber>
      <email>ghernande@classicmodelcars.com</email>
      <employeeNumber>1370</employeeNumber>
      <extension>x2028</extension>
      <firstName>Gerard</firstName>
      <jobTitle>Sales Rep</jobTitle>
      <lastName>Hernandez</lastName>
      - <officeCode>
        <addressLine1>43 Rue Jouffroy D'abbans</addressLine1>
        <city>Paris</city>
        <country>France</country>
        <officeCode>4</officeCode>
        <phone>+33 14 723 4404</phone>
        <postalCode>75017</postalCode>
        <territory>EMEA</territory>
      </officeCode>
      - <reportsTo>
        <email>gbondur@classicmodelcars.com</email>
        <employeeNumber>1102</employeeNumber>
        <extension>x5408</extension>
      </reportsTo>
    </salesRepEmployeeNumber>
  </customers>
</customers>
```



REST – JAVA

Exemplo nº2: Acesso à base de dados e devolução de registos em XML

TESTE: POSTMAN

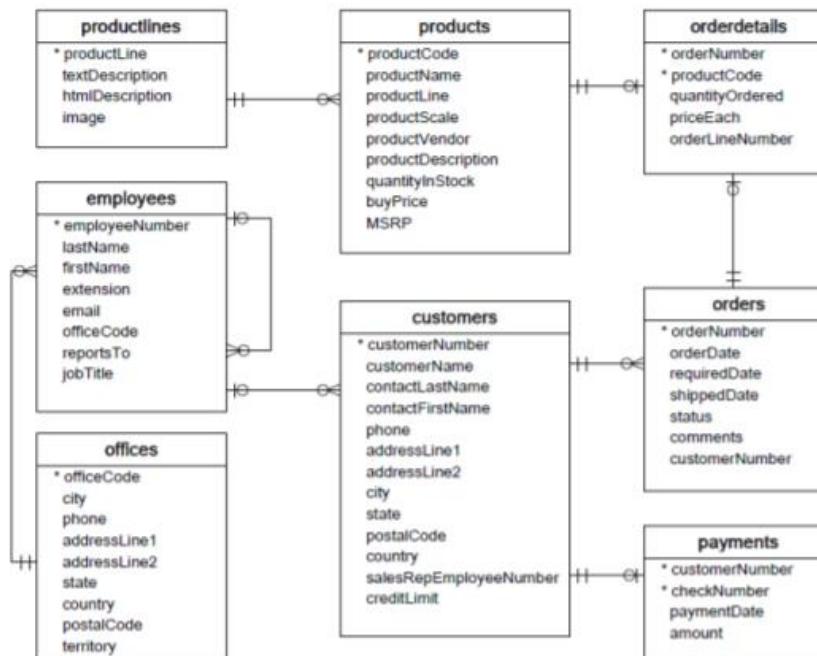
The screenshot shows the Postman interface with a GET request to `http://localhost:8080/REST_webservice_data_from_mysql/webresources/classes.customers`. The 'Params' tab is selected, showing a single parameter `wsdl`. The 'Body' tab displays the XML response:

```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <customers>
3   <customer>
4     <addressLine1>54, rue Royale</addressLine1>
5     <city>Nantes</city>
6     <contactFirstName>Carine</contactFirstName>
7     <contactLastName>Schmitz</contactLastName>
8     <country>France</country>
9     <creditLimit>21000.00</creditLimit>
10    <customerName>Atelier graphique</customerName>
11    <customerNumber>103</customerNumber>
12    <phone>+33 40 32 2555</phone>
13    <postalCode>44000</postalCode>
14    <salesRepEmployeeNumber>
15      <email>ghernande@classicmodelcars.com</email>
16      <employeeNumber>11370</employeeNumber>
17      <extension>x2028</extension>
18      <firstName>Gerard</firstName>
19      <jobTitle>Sales Rep</jobTitle>
20      <lastName>Hernandez</lastName>
21      <officeCode>
22        <addressLine1>43 Rue Jouffroy D'abbans</addressLine1>
23        <city>Paris</city>
24        <country>France</country>
25        <officeCode>4</officeCode>
26        <phone>+33 14 723 4404</phone>
27        <postalCode>75017</postalCode>
28        <territory>EMEA</territory>
29      </officeCode>
30      <reportsTo>
31        <email>gbondur@classicmodelcars.com</email>
32        <employeeNumber>11082</employeeNumber>
33      </reportsTo>
34    </salesRepEmployeeNumber>
35  </customer>
36</customers>
```

REST – PHP

Exemplo nº1: Acesso à base de dados e devolução de registos em XML

Os registos utilizados neste exemplo encontram-se disponíveis na base de dados de testes disponível em <http://www.mysqltutorial.org/mysql-sample-database.aspx>. O web service foi criado seguindo o tutorial em: <https://davidwalsh.name/web-service-php-mysql-xml-json>



<https://davidwalsh.name/web-service-php-mysql-xml-json>



REST – PHP

Exemplo nº1: Acesso à base de dados e devolução de registo em XML

Código:

```
@@ REST_get_customers_from_mysql_xml.php > ...
1  <?php
2  /* máximo de registos */
3  $number_of_posts = 10;
4  /* número de registo de inicio a listar */
5  $inicio = 1 * $number_of_posts;
6
7  /* ligar à base de dados */
8  $link = mysqli_connect('localhost','root','');
9  mysqli_select_db($link, 'classicmodels') or die('Cannot select the DB');
10
11 /* obtem dados */
12 $query = "SELECT customerNumber, customerName, phone, addressLine1, city, country FROM customers LIMIT $inicio, $number_of_posts";
13 $result = mysqli_query($link, $query) or die('Errant query: '.$query);
14
15 $arr = array();
16 if(mysqli_num_rows($result)) {
17     while($linha = mysqli_fetch_assoc($result)) {
18         $arr[] = array('customers'=>$linha);
19     }
20 }
21
22 header('Content-type: text/xml');
23 echo '<customers>'; // tag inicial do documento
24 foreach($arr as $index => $cliente) { // percorre array obtido da base de dados
25     if(is_array($cliente)) {
26         foreach($cliente as $key => $value) { // cada $cliente é um registo da base de dados representado num array de 1 elemento | obtém esse elemento para $value
27             echo '<'.$key.'>'; // inicia tag
28             if(is_array($value)) {
29                 foreach($value as $tag => $val) { // percorre todos os elementos de cada registo obtido da base de dados
30                     echo '<'.$tag.'>',htmlentities($val),'
```



REST – PHP

Exemplo nº1: Acesso à base de dados e devolução de registo em XML

TESTE: Browser (Aceder ao link do serviço)



```
Search by name or XPath

<customers>
  <customers>
    <customerNumber> 141 </customerNumber>
    <customerName> Euro+ Shopping Channel </customerName>
    <phone> (91) 555 94 44 </phone>
    <addressLine1> C/ Moralzarjal, 86 </addressLine1>
    <city> Madrid </city>
    <country> Spain </country>
  </customers>
  <customers>
    <customerNumber> 144 </customerNumber>
    <customerName> Volvo Model Replicas, Co </customerName>
    <phone> 0921-12 3555 </phone>
    <addressLine1 />
    <city />
    <country> Sweden </country>
  </customers>
  <customers>
    <customerNumber> 145 </customerNumber>
    <customerName> Danish Wholesale Imports </customerName>
    <phone> 31 12 3555 </phone>
    <addressLine1 />
    <city> Kopenhagen </city>
    <country> Denmark </country>
  </customers>
  <customers>
    <customerNumber> 146 </customerNumber>
    <customerName> Saveley & Henriot, Co. </customerName>
    <phone> 78.32.5555 </phone>
    <addressLine1> 2, rue du Commerce </addressLine1>
    <city> Lyon </city>
    <country> France </country>
  </customers>
</customers>
```



REST – PHP

Exemplo nº1: Acesso à base de dados e devolução de registo em XML

TESTE: POSTMAN

The screenshot shows the Postman application interface. The URL in the header is `http://localhost/IS_WebServices/REST_get_customers_from_mysql_xml.php`. The 'Params' tab is selected, showing two query parameters: 'wsdl' and 'Key'. The 'Body' tab is selected, showing the XML response. The XML content is as follows:

```
1 <customers>
2   <customers>
3     <customerNumber>141</customerNumber>
4     <customerName>Euro+ Shopping Channel</customerName>
5     <phone>(91) 555 94 44</phone>
6     <addressLine1>C/ Moralzarzal, 86</addressLine1>
7     <city>Madrid</city>
8     <country>Spain</country>
9   <customers>
10  <customers>
11    <customerNumber>144</customerNumber>
12    <customerName>Volvo Model Replicas, Co</customerName>
13    <phone>0921-12 3555</phone>
14    <addressLine1></addressLine1>
15    <city></city>
16    <country>Sweden</country>
17  <customers>
18  <customers>
19    <customerNumber>145</customerNumber>
20    <customerName>Danish Wholesale Imports</customerName>
21    <phone>31 12 3555</phone>
22    <addressLine1></addressLine1>
23    <city>Kopenhagen</city>
24    <country>Denmark</country>
25  <customers>
26  <customers>
27    <customerNumber>146</customerNumber>
28    <customerName>Saveley & Henriet, Co.</customerName>
29    <phone>78.32.5555</phone>
30    <addressLine1>2, rue du Commerce</addressLine1>
31    <city>Lyon</city>
32    <country>France</country>
```



REST – PHP

Exemplo nº2: Enviar conteúdo do ficheiro Books.xml

Código:

```
<?php
    $xml = file_get_contents("books.xml");
    header('Content-type: text/xml');
    echo $xml;
?>
```



REST – PHP

Exemplo nº2: Enviar conteúdo do ficheiro Books.xml

TESTE: Browser (Aceder ao link do serviço)

```
<catalog>
  <book id="bk101">
    <author> Gambardella, Matthew </author>
    <title> XML Developer's Guide </title>
    <genre> Computer </genre>
    <price> 44.95 </price>
    <publish_date> 2000-10-01 </publish_date>
    <description> An in-depth look at creating applications with XML. </description>
  </book>
  <book id="bk102">
    <author> Ralls, Kim </author>
    <title> Midnight Rain </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-12-16 </publish_date>
    <description>
      A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.
    </description>
  </book>
  <book id="bk103">
    <author> Corets, Eva </author>
    <title> Maeve Ascendant </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-11-17 </publish_date>
    <description>
      After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.
    </description>
  </book>
  <book id="bk104">
    <author> Corets, Eva </author>
    <title> Oberon's Legacy </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2001-03-10 </publish_date>
    <description>
      In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of Lor
    </description>
  </book>
</catalog>
```



REST – PHP

Exemplo nº2: Enviar conteúdo do ficheiro Books.xml

TESTE: Postman

The screenshot shows the Postman application interface. A GET request is made to the URL `http://localhost/I5_WebServices/REST_send_booksXML.php`. In the 'Params' tab, there is a single entry for 'namePesquisado' with a value of '111'. The 'Body' tab is selected, and the response is displayed in 'Pretty' format. The XML content is as follows:

```
1 <?xml version="1.0"?>
2 <catalog>
3   <book id="bk101">
4     <author>Gambardella, Matthew</author>
5     <title>XML Developer's Guide</title>
6     <genre>Computer</genre>
7     <price>44.95</price>
8     <publish_date>2000-10-01</publish_date>
9     <description>An in-depth look at creating applications
10    with XML.</description>
11  </book>
12  <book id="bk102">
13    <author>Ralls, Kim</author>
14    <title>Midnight Rain</title>
15    <genre>Fantasy</genre>
16    <price>5.95</price>
17    <publish_date>2000-12-16</publish_date>
18    <description>A former architect battles corporate zombies,
19    an evil sorceress, and her own childhood to become queen
20    of the world.</description>
21  </book>
22  <book id="bk103">
23    <author>Coresett, Eva</author>
24    <title>Maeve Ascendant</title>
25    <genre>Fantasy</genre>
26    <price>5.95</price>
27    <publish_date>2000-11-17</publish_date>
28    <description>After the collapse of a nanotechnology
29    society in England, the young survivors lay the
30    foundation for a new society.</description>
31  </book>
32  <book id="bk104">
```



JAVA – SOAP - DOM

Dados Necessários

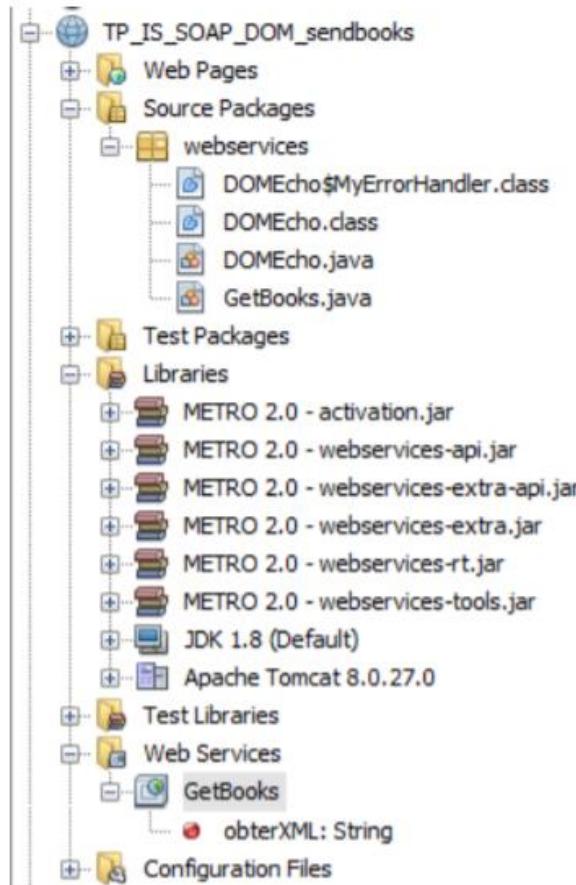
De modo a que os seguintes Web Services funcionem corretamente, é necessário:

- Necessário colocar o ficheiro books.xml em C:\testar\
- Necessário colocar o ficheiro po.xml em C:\testar\ (*apenas para os ws para o parser JAXB*)



JAVA – SOAP - DOM

Estrutura do Projeto Criado



DOMEcho.java – ficheiro que trata do parsing
DOM

GetBooks.java – classe do web service que
invoca o DOMEcho



JAVA – SOAP - DOM

Parte do Código DOMEcho.java

```
public class DOMEcho {
    static final String outputEncoding = "UTF-8";
    private PrintWriter out;
    private int indent = 0;
    private final String basicIndent = "  ";
    static final String JAXP_SCHEMA_LANGUAGE = "http://java.sun.com/xml/jaxp/properties/schemaLanguage";
    static final String W3C_XML_SCHEMA = "http://www.w3.org/2001/XMLSchema";
    static final String JAXP_SCHEMA_SOURCE = "http://java.sun.com/xml/jaxp/properties/schemaSource";

    static String a = new String();

    DOMEcho(PrintWriter out) {
        this.out = out;
    }

    private void printlnCommon(Node n) {
        a += " nodeName=\"" + n.getNodeName() + "\"";
        String val = n.getNamespaceURI();
        if (val != null) {
            a += " uri=\"" + val + "\"";
        }
        val = n.getPrefix();
        if (val != null) {
            a += " pre=\"" + val + "\"";
        }
        val = n.getLocalName();
        if (val != null) {
            a += " local=\"" + val + "\"";
        }
        val = n.getNodeValue();
        if (val != null) {
            a += " nodeValue=\"";
            if (val.trim().equals("")) {
                // Whitespace
                a += "[WS]";
            } else {
                a += "\"" + n.getNodeValue() + "\"";
            }
        }
        a += "\n";
    }

    private void outputIndentation() {
        for (int i = 0; i < indent; i++) {

```

JAVA – SOAP - DOM

Código do WS GetBooks.java

```
/*
 * 
 * @author Vasco Alves
 */
@WebService(serviceName = "GetBooks")
public class GetBooks {

    @WebMethod(operationName = "obterXML")
    public String hello() {
        try {
            return DOMEcho.obterStringXML("C:\\\\testar\\\\books.xml");
        } catch (Exception ex) {
            return "Ocorreu um erro! -> " + ex.getMessage();
        }
    }
}
```



JAVA – SOAP - DOM

Teste SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane, titled "Request 1", displays a SOAP message in XML:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope">
  <soapenv:Header>
  <soapenv:Body>
    <ns1:obterXMLResponse xmlns:ns1="http://webservices/">
      <return>DOC: nodeName="#document"
        ELEM: nodeName="catalog" local="catalog"
          TEXT: nodeName="#text" nodeValue="[NS]"
        ELEM: nodeName="book" local="book"
          ATTR: nodeName="id" local="id" nodeValue="bk101"
            TEXT: nodeName="#text" nodeValue="bk101"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="author" local="author"
            TEXT: nodeName="#text" nodeValue="Gambardella, Matthew"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="title" local="title"
            TEXT: nodeName="#text" nodeValue="XML Developer's Guide"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="genre" local="genre"
            TEXT: nodeName="#text" nodeValue="Computer"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="price" local="price"
            TEXT: nodeName="#text" nodeValue="44.95"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="publish_date" local="publish_date"
            TEXT: nodeName="#text" nodeValue="2000-10-01"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="description" local="description"
            TEXT: nodeName="#text" nodeValue="An in-depth look at creating applications with XSL."
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="book" local="book"
            ATTR: nodeName="id" local="id" nodeValue="bk102"
              TEXT: nodeName="#text" nodeValue="bk102"
              TEXT: nodeName="#text" nodeValue="[NS]"
            ELEM: nodeName="author" local="author"
              TEXT: nodeName="#text" nodeValue="Ralls, Kim"
              TEXT: nodeName="#text" nodeValue="[NS]"
            ELEM: nodeName="title" local="title"
              TEXT: nodeName="#text" nodeValue="Midnight Rain"
              TEXT: nodeName="#text" nodeValue="[NS]"
            ELEM: nodeName="genre" local="genre"
              TEXT: nodeName="#text" nodeValue="Fantasy"
              TEXT: nodeName="#text" nodeValue="[NS]"
        </return>
    </ns1:obterXMLResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

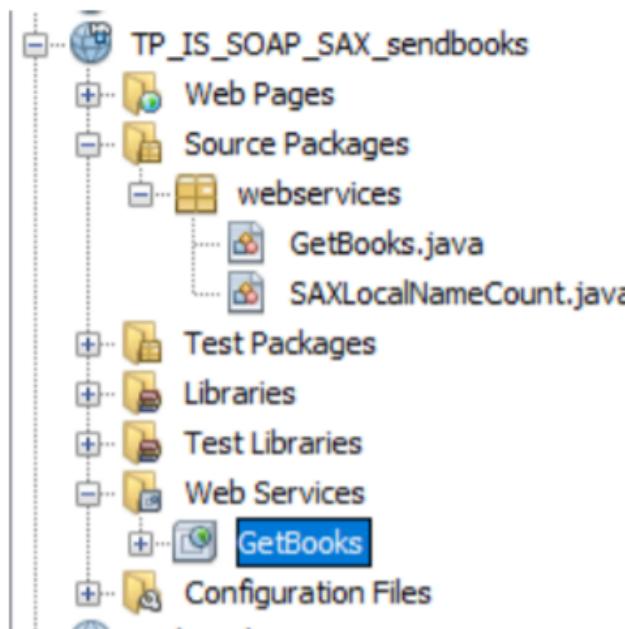
The right pane, titled "Response 1", displays the XML response received from the service:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns1:obterXMLResponse xmlns:ns1="http://webservices/">
      <return>DOC: nodeName="#document"
        ELEM: nodeName="catalog" local="catalog"
          TEXT: nodeName="#text" nodeValue="[NS]"
        ELEM: nodeName="book" local="book"
          ATTR: nodeName="id" local="id" nodeValue="bk101"
            TEXT: nodeName="#text" nodeValue="bk101"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="author" local="author"
            TEXT: nodeName="#text" nodeValue="Gambardella, Matthew"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="title" local="title"
            TEXT: nodeName="#text" nodeValue="XML Developer's Guide"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="genre" local="genre"
            TEXT: nodeName="#text" nodeValue="Computer"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="price" local="price"
            TEXT: nodeName="#text" nodeValue="44.95"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="publish_date" local="publish_date"
            TEXT: nodeName="#text" nodeValue="2000-10-01"
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="description" local="description"
            TEXT: nodeName="#text" nodeValue="An in-depth look at creating applications with XSL."
            TEXT: nodeName="#text" nodeValue="[NS]"
          ELEM: nodeName="book" local="book"
            ATTR: nodeName="id" local="id" nodeValue="bk102"
              TEXT: nodeName="#text" nodeValue="bk102"
              TEXT: nodeName="#text" nodeValue="[NS]"
            ELEM: nodeName="author" local="author"
              TEXT: nodeName="#text" nodeValue="Ralls, Kim"
              TEXT: nodeName="#text" nodeValue="[NS]"
            ELEM: nodeName="title" local="title"
              TEXT: nodeName="#text" nodeValue="Midnight Rain"
              TEXT: nodeName="#text" nodeValue="[NS]"
            ELEM: nodeName="genre" local="genre"
              TEXT: nodeName="#text" nodeValue="Fantasy"
              TEXT: nodeName="#text" nodeValue="[NS]"
        </return>
    </ns1:obterXMLResponse>
  </S:Body>
</S:Envelope>
```

At the bottom of the interface, there are tabs for "Auth", "Headers (0)", "Attachments (0)", "WS-A", "WS-RM", "JMS Headers", "JMS Property (0)", "response time: 95ms (17874 bytes)", "Headers (5)", "Attachments (0)", "SSL Info", "WSS (0)", "JMS (0)", and a progress bar.

JAVA – SOAP - SAX

Estrutura do Projeto



SAXLocalNameCount.java – ficheiro que conta o número de ocorrências dos nós XML usando o SAX

GetBooks.java – classe do web service que invoca o SAXLocalNameCount. Nesta classe é identificado o ficheiro XML a usar



JAVA – SOAP - SAX

Parte do Código de SAXLocalNameCount.java

```
public static String obterDados(String filename) throws Exception {
    boolean dtdValidate = false;
    boolean xsdValidate = false;
    String schemaSource = null;

    if (filename == null) {
        filename = "C:\\testar\\books.xml";
    }

    SAXParserFactory spf = SAXParserFactory.newInstance();
    spf.setNamespaceAware(true);
    spf.setValidating(dtdValidate || xsdValidate);
    SAXParser saxParser = spf.newSAXParser();
    if (schemaSource != null) {
        saxParser.setProperty(JAXP_SCHEMA_SOURCE, new File(schemaSource));
    }

    XMLReader xmlReader = saxParser.getXMLReader();
    xmlReader.setContentHandler(new SAXLocalNameCount());
    xmlReader.setErrorHandler(new MyErrorHandler(System.err));
    xmlReader.parse(convertToFileURL(filename));

    return teste;
}
```



JAVA – SOAP - SAX

Código do WS GetBooks.java

```
package webservices;

import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

/*
 * @author Vasco Alves
 */
@WebService(serviceName = "GetBooks")
public class GetBooks {

    @WebMethod(operationName = "obterXML")
    public String obterXML() {
        try{
            return SAXLocalNameCount.obterDados("C:\\testar\\books.xml");
        }catch(Exception e){
            return "ocorreu um erro...";
        }
    }
}
```



JAVA – SOAP - SAX

Teste SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane displays the XML request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope">
<soapenv:Header/>
<soapenv:Body>
<web:obterXML/>
</soapenv:Body>
</soapenv:Envelope>
```

The right pane displays the XML response, which includes a summary of element occurrences:

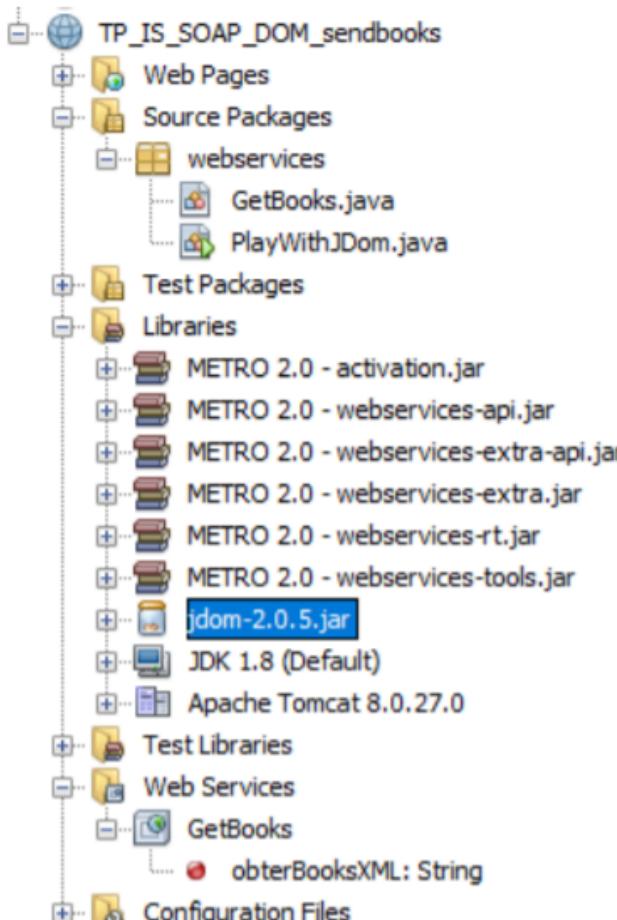
```
S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
<ns2:obterXMLResponse xmlns:ns2="http://webservices/">
<return>Local Name "genre" occurs 12 times
Local Name "catalog" occurs 1 times
Local Name "book" occurs 12 times
Local Name "author" occurs 12 times
Local Name "description" occurs 12 times
Local Name "price" occurs 12 times
Local Name "publish_date" occurs 12 times
Local Name "title" occurs 12 times</return>
</ns2:obterXMLResponse>
</S:Body>
</S:Envelope>
```

Below the panes, there are tabs for Auth, Headers (0), Attachments (0), WS-A, WS-RM, JMS Headers, JMS Property (0), Headers (5), Attachments (0), SSL Info, WSS (0), JMS (0), and a status bar indicating a response time of 55ms (600 bytes).



JAVA – SOAP - JDOM

Estrutura do Projeto



PlayWithJDom.java – ficheiro que chama o JDOM

GetBooks.java – classe do web service que invoca o PlayWithJDOM. Nesta classe é identificado o ficheiro XML a usar

Necessário adicionar jdom-2.0.5.jar às librarias do projeto



JAVA – SOAP - JDOM

Parte do Código de PlayWithJDOM.java

```
import java.io.File;
import java.io.IOException;
import java.io.*;
import org.jdom2.Document;
import org.jdom2.Element;
import org.jdom2.JDOMException;
import org.jdom2.input.SAXBuilder;
import org.jdom2.output.DOMOutputter;
import org.jdom2.output.XMLOutputter;

public class PlayWithJDom {
    XMLOutputter printer = null;

    public PlayWithJDom () {
        printer = new XMLOutputter();
    }

    public String parseAndPlay(File anXmlDocFile){
        SAXBuilder parser = new SAXBuilder();
        try {
            Document doc = parser.build(anXmlDocFile);
            return this.printJDom(doc);
        } catch (JDOMException ex) {
            ex.printStackTrace();
        } catch (IOException ex) {
            ex.printStackTrace();
        }
        return "Ocorreu um erro";
    }

    public String printJDom(Document aJDOMdoc) throws IOException{
        return printer.outputString(aJDOMdoc);
    }
}
```



JAVA – SOAP - JDOM

Código do WS GetBooks.java

```
package webservices;

import java.io.File;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

/**
 * 
 * @author Vasco Alves
 */
@WebService(serviceName = "GetBooks")
public class GetBooks {

    @WebMethod(operationName = "obterBooksXML")
    public String hello() {
        PlayWithJDom play = new PlayWithJDom();
        String XML = play.parseAndPlay(new File("C:\\testar\\books.xml"));
        return XML;
    }
}
```



JAVA – SOAP - JDOM

Teste SoapUI

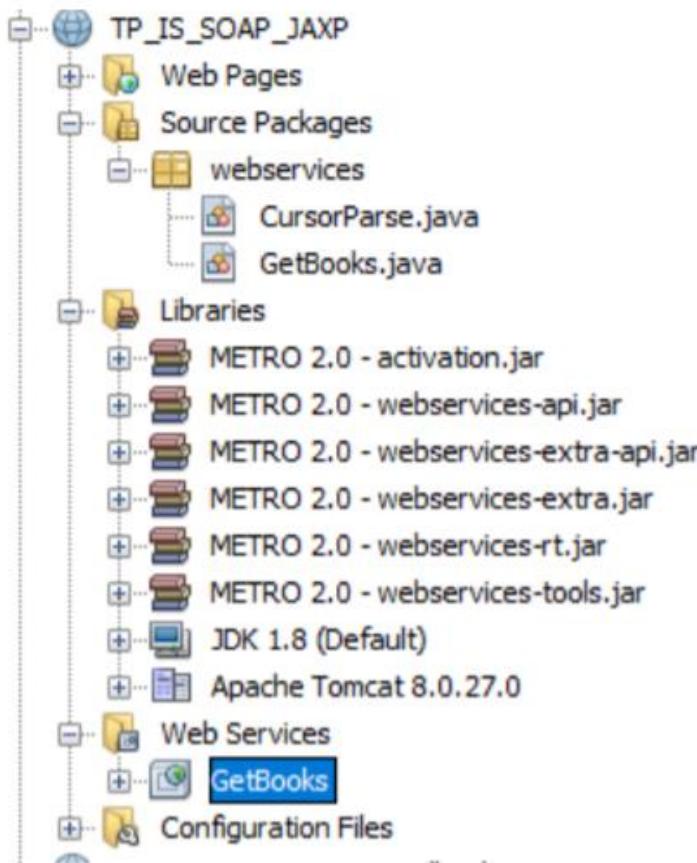
The screenshot shows the SoapUI interface with a successful SOAP interaction. The left pane displays the XML request sent to the endpoint `http://localhost:8084/TP_IS_SOAP_DOM_sendbooks/GetBooks`. The right pane shows the XML response received, which contains a catalog of books. The response XML is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<ns1:obterBooksXMLResponse xmlns:ns1="http://webservices/">
    <catalog>
        <book id="bk101">
            <author>Gambardella, Matthew</author>
            <title>XML Developer's Guide</title>
            <genre>Computer</genre>
            <price>44.95</price>
            <publish_date>2000-10-01</publish_date>
            <description>An in-depth look at creating applications with XML.</description>
        </book>
        <book id="bk102">
            <author>Ralls, Kim</author>
            <title>Midnight Rain</title>
            <genre>Fantasy</genre>
            <price>5.95</price>
            <publish_date>2000-12-16</publish_date>
            <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
        </book>
        <book id="bk103">
            <author>Corets, Eva</author>
            <title>Maeve Ascendant</title>
            <genre>Fantasy</genre>
            <price>5.95</price>
            <publish_date>2000-11-17</publish_date>
            <description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description>
        </book>
        <book id="bk104">
            <author>Corets, Eva</author>
            <title>Oberon's Legacy</title>
            <genre>Fantasy</genre>
            <price>5.95</price>
            <publish_date>2001-03-10</publish_date>
            <description>In post-apocalyptic England, the mysterious queen known only as Oberon hails to create a new life.</description>
        </book>
    </catalog>
</ns1:obterBooksXMLResponse>
```

At the bottom of the interface, there are tabs for Auth, Headers (0), Attachments (0), WS-A, WS-RM, JMS Headers, JMS Property (0), Headers (5), Attachments (0), SSL Info, WSS (0), and JMS (0). The status bar indicates a response time of 74ms (6449 bytes).

JAVA – SOAP - JAXP

Estrutura do Projeto



CursorParse.java – ficheiro que executa o parser JAXP no ficheiro XML

GetBooks.java – classe do web service que invoca o CursorParse. Nesta classe é identificado o ficheiro XML a usar



JAVA – SOAP - JAXP

Parte do Código de CursorParse.java

```
public static String obterDados(String filename) throws Exception {
    File file = null;
    try {
        if(filename == null)
            filename = "C:\\testar\\books.xml";
        file = new File(filename);
        if (!file.exists()) {
            System.out.println("File: " + filename + " does not exist.");
            System.exit(0);
        }
    } catch (ArrayIndexOutOfBoundsException aioobe){
        printUsage();
        System.exit(0);
    } catch (Exception ex){
        printUsage();
        ex.printStackTrace();
    }

    XMLInputFactory xmlif = null ;
    try{
        xmlif = XMLInputFactory.newInstance();
        xmlif.setProperty(XMLInputFactory.IS_REPLACEING_ENTITY_REFERENCES, Boolean.TRUE);
        xmlif.setProperty(XMLInputFactory.IS_SUPPORTING_EXTERNAL_ENTITIES, Boolean.FALSE);
        xmlif.setProperty(XMLInputFactory.IS_COALESCING , Boolean.FALSE);
    }catch(Exception ex){
        ex.printStackTrace();
    }
    dados += "FACTORY: " + xmlif + "\n";
    dados += "filename = "+ filename + "\n\n";
    long starttime = System.currentTimeMillis();

    try{
        XMLStreamReader xmlr = xmlif.createXMLStreamReader(filename, new FileInputStream(file));
    }
```



JAVA – SOAP - JAXP

Código do WS GetBooks.java

```
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

/**
 *
 * @author Vasco Alves
 */
@WebService(serviceName = "GetBooks")
public class GetBooks {

    @WebMethod(operationName = "obterXML")
    public String obterXML() {
        try {
            return CursorParse.obterDados("C:\\\\testar\\\\books.xml");
        } catch (Exception ex) {
            return "Ocorreu um erro...";
        }
    }
}
```



JAVA – SOAP - JAXP

Teste SoapUI

The screenshot shows the SoapUI interface with a request and response pane. The request pane contains a SOAP message with an envelope and a body. The response pane displays an XML catalog of books. The XML code is as follows:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns2:obterXMLResponse xmlns:ns2="http://webservices/*">
      <return><![CDATA[FACTORY: com.ctc.wstx.stax.WstxInputFactory@194f1ceb
filename = C:\testar\books.xml

<?xml version="1.0" encoding="null"?>

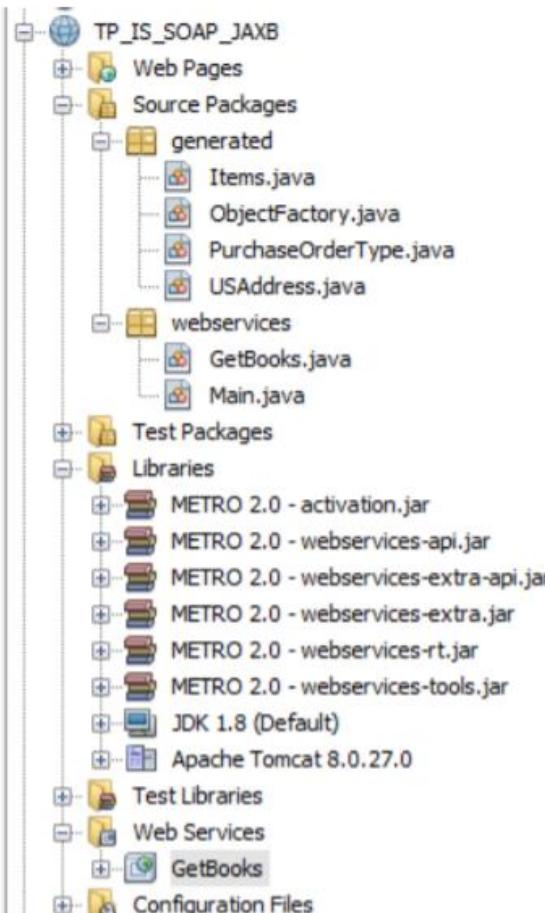
<catalog>
  <book>
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications
with XML.</description>
  </book>
  <book>
    <author>Ralls, Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-12-16</publish_date>
    <description>A former architect battles corporate zombies,
an evil sorceress, and her own childhood to become queen
of the world.</description>
  </book>
  <book>
    <author>Corets, Eva</author>
    <title>Naevia Ascendant</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-11-17</publish_date>
    <description>After the collapse of a nanotechnology
society in England, the young survivors lay the
foundation for a new society.</description>
  </book>
  <book>
    <author>Corets, Eva</author>
    <title>Oberon's Legacy</title>
    <genre>Fantasy</genre>
  </book>
</catalog>
</ns2:obterXMLResponse>
</S:Body>
</S:Envelope>
```

Auth Headers (0) Attachments (0) WS-A WS-RM JMS Headers JMS Property (0)
Headers (5) Attachments (0) SSL Info WSS (0) JMS (0)
response time: 39ms (5688 bytes)



JAVA – SOAP – JAXB

Estrutura do Projeto



Main.java – ficheiro que executa o parser JAXB no ficheiro XML

GetBooks.java – classe do web service que invoca o Main. Nesta classe é identificado o ficheiro XML a usar



JAVA – SOAP - JAXB

Parte do Código de Main.java

```
public static String obterDados(String filename) {
    File file = null;
    try {
        if(filename == null)
            filename = "C:\\\\testar\\\\po.xml";
        file = new File(filename);
        JAXBContext jc = JAXBContext.newInstance( "generated" );

        Unmarshaller u = jc.createUnmarshaller();
        JAXBElement<?> poElement = (JAXBElement<?>)u.unmarshal( new FileInputStream( file ) );
        PurchaseOrderType po = (PurchaseOrderType)poElement.getValue();

        dados += "Ship the following items to: \\n";
        USAddress address = po.getShipTo();
        displayAddress( address );

        Items items = po.getItems();
        displayItems(items);

        return dados;
    } catch( JAXBException je ) {
        return "Ocorreu um erro...";
    } catch( IOException ioe ) {
        return "Ocorreu um erro...";
    }
}
```



JAVA – SOAP - JAXB

Código do WS GetBooks.java

```
package webservices;

import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.jws.WebParam;

/**
 * 
 * @author Vasco Alves
 */
@WebService(serviceName = "GetBooks")
public class GetBooks {

    @WebMethod(operationName = "obterXML")
    public String obterXML() {
        return Main.obterDados("C:\\testar\\po.xml");
    }
}
```



JAVA – SOAP – JAXB

Teste SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane displays the XML request sent to the service, and the right pane displays the XML response received.

Request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope">
  <soapenv:Header/>
  <soapenv:Body>
    <ns1:obterXML/>
  </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Body>
    <ns1:obterXMLResponse xmlns:ns1="http://wsahservices/">
      <return>Ship the following items to:
          Alice Smith
          123 Maple Street
          Cambridge, MA 12345
          US

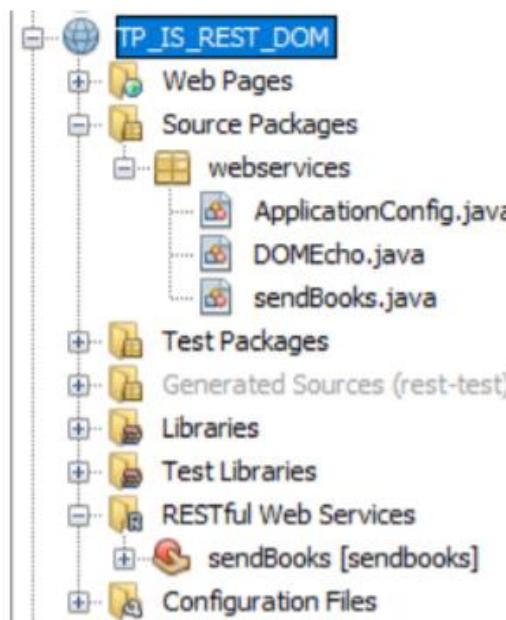
      Items:
          5 copies of "Nosferatu - Special Edition (1922)"
          3 copies of "The Mummy (1969)"
          3 copies of "Godzilla and Mothra: Battle for Earth/Godzilla vs. King Ghidorah"
    </ns1:obterXMLResponse>
  </S:Body>
</S:Envelope>
```

At the bottom of the SoapUI interface, there are tabs for Auth, Headers (0), Attachments (0), WS-A, WS-RM, JMS Headers, JMS Property (0) on the left, and Headers (5), Attachments (0), SSL Info, WSS (0), JMS (0) on the right. The status bar at the bottom indicates a response time of 42ms (513 bytes).



JAVA – REST - DOM

Estrutura do Projeto Criado



DOMEcho.java – ficheiro que trata do parsing
DOM

GetBooks.java – classe do web service que
invoca o DOMEcho



JAVA – REST - DOM

Parte do Código DOMEcho.java

```
public class DOMEcho {  
    static final String outputEncoding = "UTF-8";  
    private PrintWriter out;  
    private int indent = 0;  
    private final String basicIndent = " ";  
    static final String JAXP_SCHEMA_LANGUAGE = "http://java.sun.com/xml/jaxp/properties/schemaLanguage";  
    static final String W3C_XML_SCHEMA = "http://www.w3.org/1001/XMLSchema";  
    static final String JAXP_SCHEMA_SOURCE = "http://java.sun.com/xml/jaxp/properties/schemaSource";  
  
    static String a = new String();  
  
    DOMEcho(PrintWriter out) {  
        this.out = out;  
    }  
  
    private void printlnCommon(Node n) {  
        a += " nodeName=\"" + n.getNodeName() + "\"";  
        String val = n.getNamespaceURI();  
        if (val != null) {  
            a += " uri=\"" + val + "\"";  
        }  
        val = n.getPrefix();  
        if (val != null) {  
            a += " pre=\"" + val + "\"";  
        }  
        val = n.getLocalName();  
        if (val != null) {  
            a += " local=\"" + val + "\"";  
        }  
        val = n.getNodeValue();  
        if (val != null) {  
            a += " nodeValue=";  
            if (val.trim().equals("")) {  
                // Whitespace  
                a += "[WS]";  
            } else {  
                a += "\n" + n.getNodeValue() + "\n";  
            }  
        }  
        a += "\n";  
    }  
  
    private void outputIndentation() {  
        for (int i = 0; i < indent; i++) {  
            out.print(basicIndent);  
        }  
    }  
}
```



JAVA – REST - DOM

Código do WS sendBooks.java

```
package webservices;

import javax.ws.rs.core.Context;
import javax.ws.rs.core.UriInfo;
import javax.ws.rs.Produces;
import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PUT;
import javax.ws.rs.core.MediaType;

@Path("sendbooks")
public class sendBooks {

    @Context
    private UriInfo context;

    public sendBooks() {}

    @GET
    @Produces(MediaType.APPLICATION_XML)
    public String getXml() {
        try {
            return DOMEcho.obterStringXML("C:\\testar\\books.xml");
        } catch (Exception ex) {
            return "Ocorreu um erro! -> " + ex.getMessage();
        }
    }
}
```



JAVA – REST - DOM

Teste Browser



```
DOC: nodeName="#document"
ELEM: nodeName="catalog" local="catalog"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="book" local="book"
ATTR: nodeName="id" local="id" nodeValue="bk101"
TEXT: nodeName="#text" nodeValue="bk101"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="author" local="author"
TEXT: nodeName="#text" nodeValue="Gambardella, Matthew"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="title" local="title"
TEXT: nodeName="#text" nodeValue="XML Developer's Guide"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="genre" local="genre"
TEXT: nodeName="#text" nodeValue="Computer"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="price" local="price"
TEXT: nodeName="#text" nodeValue="44.95"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="publish_date" local="publish_date"
TEXT: nodeName="#text" nodeValue="2000-10-01"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="description" local="description"
TEXT: nodeName="#text" nodeValue="An in-depth look at creating applications
with XML."
TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="book" local="book"
ATTR: nodeName="id" local="id" nodeValue="bk102"
TEXT: nodeName="#text" nodeValue="bk102"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="author" local="author"
TEXT: nodeName="#text" nodeValue="Ralls, Kim"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="title" local="title"
TEXT: nodeName="#text" nodeValue="Midnight Rain"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="genre" local="genre"
```



JAVA – REST - DOM

Teste POSTMAN

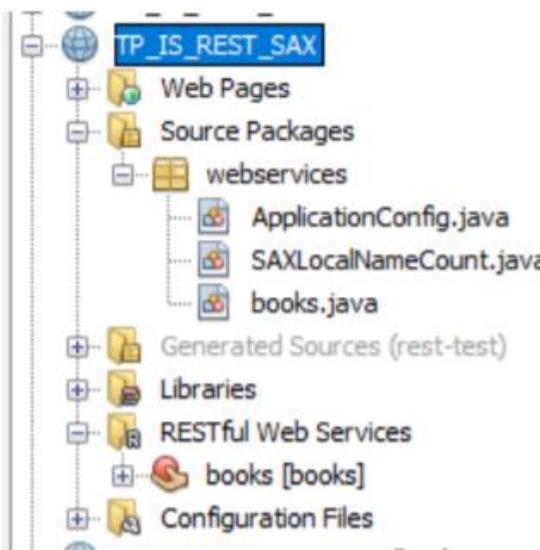
The screenshot shows the POSTMAN interface with the following details:

- Untitled Request**
- Method:** GET
- URL:** http://localhost:8084/TP_IS_REST_DOM/webservices/sendbooks
- Params:** Key
- Body:** PRETTY
- Response Body (Pretty Print):**

```
1 DOC: nodeName="#document"
2 ELEM: nodeName="catalog" local="catalog"
3 TEXT: nodeName="#text" nodeValue=[WS]
4 ELEM: nodeName="book" local="book"
5 ATTR: nodeName="id" local="id" nodeValue="bk101"
6 TEXT: nodeName="#text" nodeValue="bk101"
7 TEXT: nodeName="#text" nodeValue=[WS]
8 ELEM: nodeName="author" local="author"
9 TEXT: nodeName="#text" nodeValue="Gambardella, Matthew"
10 TEXT: nodeName="#text" nodeValue=[WS]
11 ELEM: nodeName="title" local="title"
12 TEXT: nodeName="#text" nodeValue="XML Developer's Guide"
13 TEXT: nodeName="#text" nodeValue=[WS]
14 ELEM: nodeName="genre" local="genre"
15 TEXT: nodeName="#text" nodeValue="Computer"
16 TEXT: nodeName="#text" nodeValue=[WS]
17 ELEM: nodeName="price" local="price"
18 TEXT: nodeName="#text" nodeValue="44.95"
19 TEXT: nodeName="#text" nodeValue=[WS]
20 ELEM: nodeName="publish_date" local="publish_date"
21 TEXT: nodeName="#text" nodeValue="2000-10-01"
22 TEXT: nodeName="#text" nodeValue=[WS]
23 ELEM: nodeName="description" local="description"
24 TEXT: nodeName="#text" nodeValue="An in-depth look at creating applications
|    with XML."
25 |
26 TEXT: nodeName="#text" nodeValue=[WS]
27 TEXT: nodeName="#text" nodeValue=[WS]
28 ELEM: nodeName="book" local="book"
29 ATTR: nodeName="id" local="id" nodeValue="bk102"
30 TEXT: nodeName="#text" nodeValue="bk102"
31 TEXT: nodeName="#text" nodeValue=[WS]
32 ELEM: nodeName="author" local="author"
33 TEXT: nodeName="#text" nodeValue="Ralls, Kim"
34 TEXT: nodeName="#text" nodeValue=[WS]
```

JAVA – REST - SAX

Estrutura do Projeto



SAXLocalNameCount.java – ficheiro que conta o número de ocorrências dos nós XML usando o SAX

GetBooks.java – classe do web service que invoca o SAXLocalNameCount. Nesta classe é identificado o ficheiro XML a usar



JAVA – REST - SAX

Parte do Código de SAXLocalNameCount.java

```
public static String obterDados(String filename) throws Exception {
    boolean dtdValidate = false;
    boolean xsdValidate = false;
    String schemaSource = null;

    if (filename == null) {
        filename = "C:\\testar\\books.xml";
    }

    SAXParserFactory spf = SAXParserFactory.newInstance();
    spf.setNamespaceAware(true);
    spf.setValidating(dtdValidate || xsdValidate);
    SAXParser saxParser = spf.newSAXParser();
    if (schemaSource != null) {
        saxParser.setProperty(JAXP_SCHEMA_SOURCE, new File(schemaSource));
    }

    XMLReader xmlReader = saxParser.getXMLReader();
    xmlReader.setContentHandler(new SAXLocalNameCount());
    xmlReader.setErrorHandler(new MyErrorHandler(System.err));
    xmlReader.parse(convertToFileURL(filename));

    return teste;
}
```



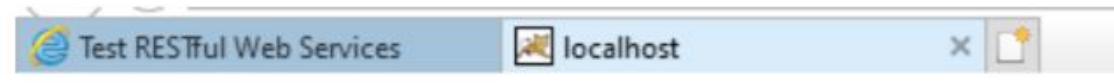
JAVA – REST - SAX

Código do WS books.java

```
1  package webservices;
2
3  import javax.ws.rs.core.Context;
4  import javax.ws.rs.core.UriInfo;
5  import javax.ws.rs.Produces;
6  import javax.ws.rs.Consumes;
7  import javax.ws.rs.GET;
8  import javax.ws.rs.Path;
9  import javax.ws.rs.PUT;
10 import javax.ws.rs.core.MediaType;
11
12 @Path("books")
13 public class books {
14
15     @Context
16     private UriInfo context;
17
18     public books() {}
19
20     @GET
21     @Produces(MediaType.TEXT_PLAIN)
22     public String getXml() {
23         try{
24             return SAXLocalNameCount.obterDados("C:\\testar\\books.xml");
25         }catch(Exception e){
26             return "ocorreu um erro...";
27         }
28     }
29 }
```

JAVA – REST - SAX

Teste Browser



```
Local Name "genre" occurs 12 times
Local Name "catalog" occurs 1 times
Local Name "book" occurs 12 times
Local Name "author" occurs 12 times
Local Name "description" occurs 12 times
Local Name "price" occurs 12 times
Local Name "publish_date" occurs 12 times
Local Name "title" occurs 12 times
Local Name "genre" occurs 12 times
Local Name "catalog" occurs 1 times
Local Name "book" occurs 12 times
Local Name "author" occurs 12 times
Local Name "description" occurs 12 times
Local Name "price" occurs 12 times
Local Name "publish_date" occurs 12 times
Local Name "title" occurs 12 times
```



JAVA – REST - SAX

Teste POSTMAN

Untitled Request

GET http://localhost:8084/TP_IS_REST_SAX/webresources/books

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results

Status: 200 OK Time: 116ms Size: 424 B Save Response ▾

Pretty Raw Preview Visualize BETA Text ⌂

```
1 Local Name "genre" occurs 12 times
2 Local Name "catalog" occurs 1 times
3 Local Name "book" occurs 12 times
4 Local Name "author" occurs 12 times
5 Local Name "description" occurs 12 times
6 Local Name "price" occurs 12 times
7 Local Name "publish_date" occurs 12 times
8 Local Name "title" occurs 12 times
9
```



JAVA – REST - JDOM

Estrutura do Projeto



PlayWithJDom.java – ficheiro que chama o JDOM

GetBooks.java – classe do web service que invoca o PlayWithJDOM. Nesta classe é identificado o ficheiro XML a usar

Necessário adicionar jdom-2.0.5.jar às librarias do projeto



JAVA – REST - JDOM

Parte do Código de PlayWithJDOM.java

```
import java.io.File;
import java.io.IOException;
import java.io.*;
import org.jdom2.Document;
import org.jdom2.Element;
import org.jdom2.JDOMException;
import org.jdom2.input.SAXBuilder;
import org.jdom2.output.DOMOutputter;
import org.jdom2.output.XMLOutputter;

public class PlayWithJDom {
    XMLOutputter printer = null;

    public PlayWithJDom () {
        printer = new XMLOutputter();
    }

    public String parseAndPlay(File anXmlDocFile){
        SAXBuilder parser = new SAXBuilder();
        try {
            Document doc = parser.build(anXmlDocFile);
            return this.printJDom(doc);
        } catch (JDOMException ex) {
            ex.printStackTrace();
        } catch (IOException ex) {
            ex.printStackTrace();
        }
        return "Ocorreu um erro";
    }

    public String printJDom(Document aJDOMdoc) throws IOException{
        return printer.outputString(aJDOMdoc);
    }
}
```



JAVA – REST - JDOM

Código do WS books.java

```
1 package webservices;
2
3 import java.io.File;
4 import javax.ws.rs.core.Context;
5 import javax.ws.rs.core.UriInfo;
6 import javax.ws.rs.Produces;
7 import javax.ws.rs.Consumes;
8 import javax.ws.rs.GET;
9 import javax.ws.rs.Path;
10 import javax.ws.rs.PUT;
11 import javax.ws.rs.core.MediaType;
12
13 @Path("books")
14 public class books {
15
16     @Context
17     private UriInfo context;
18
19     public books() {}
20
21     @GET
22     @Produces(MediaType.APPLICATION_XML)
23     public String getXml() {
24         PlayWithJDom play = new PlayWithJDom();
25         String XML = play.parseAndPlay(new File("C:\\testar\\books.xml"));
26         return XML;
27     }
28 }
29
```



JAVA – REST - JDOM

Teste Browser



```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
- <book id="bk101">
  <author>Gambardella, Matthew</author>
  <title>XML Developer's Guide</title>
  <genre>Computer</genre>
  <price>44.95</price>
  <publish_date>2000-10-01</publish_date>
  <description>An in-depth look at creating applications with XML.</description>
</book>
- <book id="bk102">
  <author>Rails, Kim</author>
  <title>Midnight Rain</title>
  <genre>Fantasy</genre>
  <price>5.95</price>
  <publish_date>2000-12-16</publish_date>
  <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
</book>
- <book id="bk103">
  <author>Corets, Eva</author>
  <title>Maeve Ascendant</title>
  <genre>Fantasy</genre>
  <price>5.95</price>
  <publish_date>2000-11-17</publish_date>
  <description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description>
</book>
- <book id="bk104">
  <author>Corets, Eva</author>
  <title>Oberon's Legacy</title>
  <genre>Fantasy</genre>
  <price>5.95</price>
  <publish_date>2001-03-10</publish_date>
  <description>In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Maeve Ascendant.</description>
</book>
- <book id="bk105">
  <author>Corets, Eva</author>
  <title>The Sundered Grail</title>
  <genre>Fantasy</genre>
  <price>5.95</price>
  <publish_date>2001-09-10</publish_date>
  <description>The two daughters of Maeve, half-sisters, battle one another for control of England. Sequel to Oberon's Legacy.</description>
</book>
- <book id="bk106">
  <author>Randall, Cynthia</author>
  <title>Lover Birds</title>
  <genre>Romance</genre>
  <price>4.95</price>
  <publish_date>2000-09-02</publish_date>
  <description>When Carla meets Paul at an ornithology conference, tempers fly as feathers get ruffled.</description>
</book>
- <book id="bk107">
  <author>Thurman, Paula</author>
  <title>Splash Splash</title>
  <genre>Romance</genre>
  <price>4.95</price>
  <publish_date>2000-11-02</publish_date>
  <description>A deep sea diver finds true love twenty thousand leagues beneath the sea.</description>
</book>
```



JAVA – REST - JDOM

Teste POSTMAN

Untitled Request

GET http://localhost:8084/TP_IS_REST_JDOM/webresources/books

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	*** Bulk Edit
Key	Value	Description	

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize BETA XML

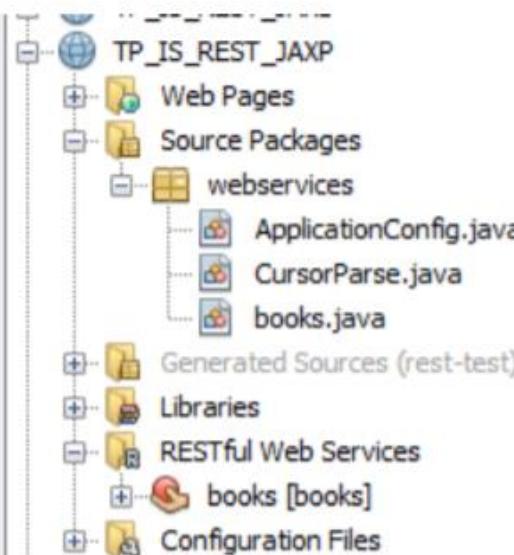
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <catalog>
3   <book id="bk101">
4     <author>Gambarella, Matthew</author>
5     <title>XML Developer's Guide</title>
6     <genre>Computer</genre>
7     <price>44.95</price>
8     <publish_date>2000-10-01</publish_date>
9     <description>An in-depth look at creating applications
10    with XML.</description>
11  </book>
12  <book id="bk102">
13    <author>Ralls, Kim</author>
14    <title>Midnight Rain</title>
15    <genre>Fantasy</genre>
16    <price>5.95</price>
17    <publish_date>2000-12-16</publish_date>
18    <description>A former architect battles corporate zombies,
19    an evil sorceress, and her own childhood to become queen
20    of the world.</description>
21  </book>
22  <book id="bk103">
23    <author>Corets, Eva</author>
24    <title>Heave Ascendant</title>
25    <genre>Fantasy</genre>
26    <price>5.95</price>
27    <publish_date>2000-11-17</publish_date>
28    <description>After the collapse of a nanotechnology
29    society in England, the young survivors lay the
30    foundation for a new society.</description>
31  </book>
32  <book id="bk104">
```

Status: 200 OK Time: 16ms Size: 4.59 KB Save Response



JAVA – REST - JAXP

Estrutura do Projeto



CursorParse.java – ficheiro que executa o parser JAXP no ficheiro XML

GetBooks.java – classe do web service que invoca o CursorParse. Nesta classe é identificado o ficheiro XML a usar



JAVA – REST - JAXP

Parte do Código de CursorParse.java

```
public static String obterDados(String filename) throws Exception {
    File file = null;
    try {
        if(filename == null)
            filename = "C:\\testar\\\\books.xml";
        file = new File(filename);
        if (!file.exists()) {
            System.out.println("File: " + filename + " does not exist.");
            System.exit(0);
        }
    } catch (ArrayIndexOutOfBoundsException aioobe){
        printUsage();
        System.exit(0);
    } catch (Exception ex){
        printUsage();
        ex.printStackTrace();
    }

    XMLInputFactory xmlif = null ;
    try{
        xmlif = XMLInputFactory.newInstance();
        xmlif.setProperty(XMLInputFactory.IS_REPLACEING_ENTITY_REFERENCES,Boolean.TRUE);
        xmlif.setProperty(XMLInputFactory.IS_SUPPORTING_EXTERNAL_ENTITIES,Boolean.FALSE);
        xmlif.setProperty(XMLInputFactory.IS_COALESCING , Boolean.FALSE);
    }catch(Exception ex){
        ex.printStackTrace();
    }
    dados += "FACTORY: " + xmlif + "\n";
    dados += "filename = "+ filename + "\n\n";
    long starttime = System.currentTimeMillis();

    try{
        XMLStreamReader xmlr = xmlif.createXMLStreamReader(filename, new FileInputStream(file));
        
```



JAVA – REST - JAXP

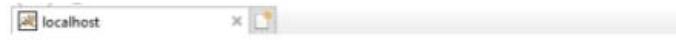
Código do WS books.java

```
1  package webservices;
2
3  import javax.ws.rs.core.Context;
4  import javax.ws.rs.core.UriInfo;
5  import javax.ws.rs.Produces;
6  import javax.ws.rs.Consumes;
7  import javax.ws.rs.GET;
8  import javax.ws.rs.Path;
9  import javax.ws.rs.PUT;
10 import javax.ws.rs.core.MediaType;
11
12 /**
13  * REST Web Service
14  *
15  * @author Vasco Alves
16  */
17 @Path("books")
18 public class books {
19
20     @Context
21     private UriInfo context;
22
23     public books() {}
24
25     @GET
26     @Produces(MediaType.TEXT_PLAIN)
27     public String getText() {
28         try {
29             return CursorParse.obterDados("C:\\testar\\\\books.xml");
30         } catch (Exception ex) {
31             return "Ocorreu um erro...";
32         }
33     }
34 }
```



JAVA – REST - JAXP

Teste Browser



```
localhost
FACTORY: com.sun.xml.internal.stream.XMLInputFactoryImpl@1e6540ec
filename = C:\testar\books.xml

<?xml version="1.0" encoding="null"?>
<catalog>
  <book>
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications
with XML.</description>
  </book>
  <book>
    <author>Ralls, Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-12-16</publish_date>
    <description>A former architect battles corporate zombies,
an evil sorceress, and her own childhood to become queen
of the world.</description>
  </book>
  <book>
    <author>Corets, Eva</author>
    <title>Maeve Ascendant</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-11-17</publish_date>
    <description>After the collapse of a nanotechnology
society in England, the young survivors lay the
foundation for a new society.</description>
  </book>
  <book>
    <author>Corets, Eva</author>
    <title>Oberon's Legacy</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2001-03-10</publish_date>
    <description>In post-apocalypse England, the mysterious
agent known only as Oberon helps to create a new life
for the inhabitants of London. Sequel to Maeve
Ascendant.</description>
  </book>
  <book>
    <author>Corets, Eva</author>
    <title>The Sundered Grail</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2001-03-10</publish_date>
    <description>The final book in the Oberon series.
When Oberon disappears, the people of London are
left to fend for themselves against the
corporate zombies and the evil sorceress. The
sequel to Oberon's Legacy and Maeve Ascendant.
  </book>

```



JAVA – REST - JAXP

Teste POSTMAN

The screenshot shows the POSTMAN interface with the following details:

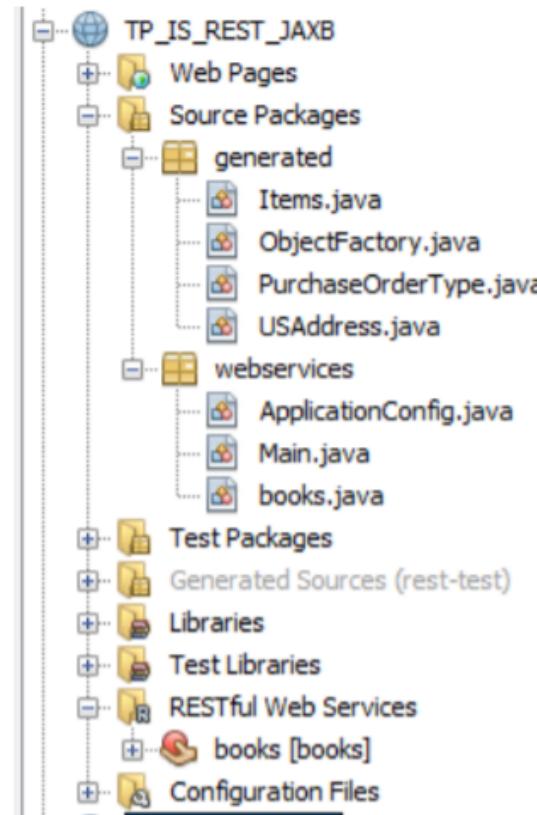
- Request Method:** GET
- URL:** http://localhost:8084/TP_IS_REST_JAXP/webservices/books
- Params Tab:** Selected. Shows a table for "Query Params" with one entry: Key (empty) and Value (empty).
- Body Tab:** Selected. Shows the XML response body.
- XML Response Body:**

```
1 FACTORY: com.sun.xml.internal.stream.XMLInputFactoryImpl@1e6548ec
2 filename = C:\testar\books.xml
3
4 <?xml version="1.0" encoding="null"?>
5 <catalog>
6   <book>
7     <author>Gambardella, Matthew</author>
8     <title>XML Developer's Guide</title>
9     <genre>Computer</genre>
10    <price>44.95</price>
11    <publish_date>2000-10-01</publish_date>
12    <description>An in-depth look at creating applications
13      with XML.</description>
14  </book>
15  <book>
16    <author>Ralls, Kim</author>
17    <title>Midnight Rain</title>
18    <genre>Fantasy</genre>
19    <price>5.95</price>
20    <publish_date>2000-12-16</publish_date>
21    <description>A former architect battles corporate zombies,
22      an evil sorceress, and her own childhood to become queen
23      of the world.</description>
24  </book>
25  <book>
26    <author>Corets, Eva</author>
27    <title>Naive Ascendant</title>
28    <genre>Fantasy</genre>
29    <price>5.95</price>
30    <publish_date>2000-11-17</publish_date>
31    <description>After the collapse of a nanotechnology
32      society in England, the young survivors lay the
33      foundation for a new society.</description>
```
- Status Bar:** Status: 200 OK, Time: 29ms, Size: 17.43 KB



JAVA – REST - JAXB

Estrutura do Projeto



Main.java – ficheiro que executa o parser JAXB no ficheiro XML

GetBooks.java – classe do web service que invoca o Main. Nesta classe é identificado o ficheiro XML a usar



JAVA – REST - JAXB

Parte do Código de Main.java

```
public static String obterDados(String filename) {
    File file = null;
    try {
        if(filename == null)
            filename = "C:\\\\testar\\\\po.xml";
        file = new File(filename);
        JAXBContext jc = JAXBContext.newInstance( "generated" );

        Unmarshaller u = jc.createUnmarshaller();
        JAXBElement<?> poElement = (JAXBElement<?>)u.unmarshal( new FileInputStream( file ) );
        PurchaseOrderType po = (PurchaseOrderType)poElement.getValue();

        dados += "Ship the following items to: \\n";

        USAddress address = po.getShipTo();
        displayAddress( address );

        Items items = po.getItems();
        displayItems(items);

        return dados;
    } catch( JAXBException je ) {
        return "Ocorreu um erro...";
    } catch( IOException ioe ) {
        return "Ocorreu um erro...";
    }
}
```



JAVA – REST - JAXB

Código do WS books.java

```
1 package webservices;
2
3 import javax.ws.rs.core.Context;
4 import javax.ws.rs.core.UriInfo;
5 import javax.ws.rs.Produces;
6 import javax.ws.rs.Consumes;
7 import javax.ws.rs.GET;
8 import javax.ws.rs.Path;
9 import javax.ws.rs.PUT;
10 import javax.ws.rs.core.MediaType;
11
12 /**
13 * REST Web Service
14 *
15 * @author Vasco Alves
16 */
17 @Path("books")
18 public class books {
19
20     @Context
21     private UriInfo context;
22
23     public books() {}
24
25     @GET
26     @Produces(MediaType.TEXT_PLAIN)
27     public String getText() {
28         try {
29             return CursorParse.obterDados("C:\\\\testar\\\\books.xml");
30         } catch (Exception ex) {
31             return "Ocorreu um erro...";
32         }
33     }
34 }
```



JAVA – REST - JAXB

Teste Browser

```
localhost

Ship the following items to:
    Alice Smith
    123 Maple Street
    Cambridge, MA 12345
    US

Items:
    5 copies of "Nosferatu - Special Edition (1929)"
    3 copies of "The Mummy (1959)"
    3 copies of "Godzilla and Mothra: Battle for Earth/Godzilla vs. King Ghidora"
Ship the following items to:
    Alice Smith
    123 Maple Street
    Cambridge, MA 12345
    US

Items:
    5 copies of "Nosferatu - Special Edition (1929)"
    3 copies of "The Mummy (1959)"
    3 copies of "Godzilla and Mothra: Battle for Earth/Godzilla vs. King Ghidora"
```



JAVA – REST - JAXB

Teste POSTMAN

The screenshot shows the Postman interface with the following details:

- Title:** Untitled Request
- Method:** GET
- URL:** http://localhost:8084/TP_JS_REST_JAXB/webresources/books
- Headers:** (7)
- Body:** (Raw, Preview, Visualize BETA, Text, JSON)
- Params:** (Query Params table)

KEY	VALUE	DESCRIPTION
Key	Value	Description
- Tests:** (Empty)
- Settings:** (Empty)
- Cookies:** (Empty)
- Code:** (Empty)
- Status:** 200 OK | Time: 159ms | Size: 387 B | Save Response ▾

The Body section displays the following JSON payload:

```
1 Ship the following items to:  
2   Alice Smith  
3     123 Maple Street  
4     Cambridge, MA 12345  
5     US  
6  
7 Items:  
8   5 copies of "Nosferatu - Special Edition (1929)"  
9   3 copies of "The Mummy (1959)"  
10  3 copies of "Godzilla and Mothra: Battle for Earth/Godzilla vs. King Ghidorah"
```

PHP – SOAP - DOM

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
dom	Pasta de ficheiros	
sax	Pasta de ficheiros	
books.xml	Documento XML	5 KB
nusoap.php	Ficheiro PHP	347 KB
pag_clienteSOAP.php	Ficheiro PHP	1 KB
servicoSOAP.php	Ficheiro PHP	1 KB
simple.xml	Documento XML	2 KB

servicoSOAP.php – ficheiro que trata de ler o ficheiro XML com o DOM e de o enviar ao cliente num envelope SOAP

pag_ClienteSOAP.php – cliente SOAP em php que chama o servicoSOAP.php



PHP – SOAP - DOM

Parte do Código serviçoSOAP.php

```
1 <?php
2     require_once "nusoap.php";
3     function obterDados($nomeFicheiroACorrer) {
4         // -- verifica se shell_exec está ativo
5         if(!function_exists('shell_exec')) {
6             die("A função shell_exec deve estar ativa!");
7         }
8
9         $output = shell_exec('java dom/DOMEcho ' . $nomeFicheiroACorrer);
10        $XML = "";
11
12        $XML = "DOC: nodeName=" . explode('DOC: nodeName=', $output)[1];
13        $XML = explode(' Parsing Time = ', $XML)[0];
14
15        return $XML;
16
17    }
18
19    // -- cria servidor SOAP e envia
20    $server = new soap_server();
21    $server->configureWSdl('AddService', 'obtemXML');
22    $server->wsdl->schemaTargetNamespace = 'obtemXML';
23    $server->register("obterDados",
24        array('tipo' => "xsd:string"),
25        array('return' => 'xsd:string')
26    );
27    $server->service(file_get_contents('php://input'));
28 ?>
```



PHP – SOAP - DOM

Código do pag_clienteSOAP.php

```
1 <?php
2 require_once "nusoap.php";
3
4 $client = new nusoap_client("http://localhost/IS_WebServices/TP_DOM_SOAP_sendbooks/servicoSOAP.php");
5
6
7 $error = $client->getError();
8 if ($error) {
9     echo "<h2>Constructor error</h2><pre>" . $error . "</pre>";
10 }
11
12 $result = $client->call("obterDados", array("nome" => "books.xml"));
13
14 if ($client->fault) {
15     echo "<h2>Fault</h2><pre>" . $result . "</pre>";
16 } else {
17     $error = $client->getError();
18     if ($error) {
19         echo "<h2>Error</h2><pre>" . $error . "</pre>";
20     } else {
21         header("content-type: text/plain");
22         echo $result;
23     }
24 }
```



PHP – SOAP - DOM

Teste SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane displays the XML request sent to the service, and the right pane displays the XML response received from the service.

Request:

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
    <soapenv:Header/>
    <soapenv:Body>
        <obterDados soapenv:encodingStyle="http://schemas.xmlsoap.org/encoding/">
            <tipo xsi:type="xsd:string">books.xml</tipo>
        </obterDados>
    </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Body>
        <ns1:obterDadosResponse xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/">
            <return xsi:type="xsd:string">DOC: nodeName="#document"
                ELEM: nodeName="catalog" local="catalog"
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="book" local="book"
                    ATTR: nodeName="id" local="id" nodeValue="bk101"
                        TEXT: nodeName="#text" nodeValue="bk101"
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="author" local="author"
                    TEXT: nodeName="#text" nodeValue=Gambardella, Matthew
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="title" local="title"
                    TEXT: nodeName="#text" nodeValue=XML Developer's Guide
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="genre" local="genre"
                    TEXT: nodeName="#text" nodeValue=Computer
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="price" local="price"
                    TEXT: nodeName="#text" nodeValue=44.95
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="publish_date" local="publish_date"
                    TEXT: nodeName="#text" nodeValue=2000-10-01
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="description" local="description"
                    TEXT: nodeName="#text" nodeValue=An in-depth look at creating applications with XML.
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="book" local="book"
                    ATTR: nodeName="id" local="id" nodeValue="bk102"
                        TEXT: nodeName="#text" nodeValue="bk102"
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="author" local="author"
                    TEXT: nodeName="#text" nodeValue=Ralls, Kim
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="title" local="title"
                    TEXT: nodeName="#text" nodeValue=Midnight Rain
                    TEXT: nodeName="#text" nodeValue=[WS]
                ELEM: nodeName="genre" local="genre"
                    TEXT: nodeName="#text" nodeValue=Fantasy
                    TEXT: nodeName="#text" nodeValue=[WS]
            </return>
        </ns1:obterDadosResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Below the panes, there are tabs for Headers (10), Attachments (0), SSL Info, WSS (0), JMS (0), and a status bar indicating a response time of 763ms (20052 bytes).



PHP – SOAP - DOM

Teste Browser (pag_clienteSOAP.php)

```
DOC: nodeName="#document"
ELEM: nodeName="catalog" local="catalog"
TEXT: nodeName="#text" nodeValue="[WS]
ELEM: nodeName="book" local="book"
ATTR: nodeName="id" local="id" nodeValue="bk101"
TEXT: nodeName="#text" nodeValue="bk101"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="author" local="author"
TEXT: nodeName="#text" nodeValue="Gambardella, Matthew"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="title" local="title"
TEXT: nodeName="#text" nodeValue="XML Developer's Guide"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="genre" local="genre"
TEXT: nodeName="#text" nodeValue="Computer"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="price" local="price"
TEXT: nodeName="#text" nodeValue="44.95"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="publish_date" local="publish_date"
TEXT: nodeName="#text" nodeValue="2000-10-01"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="description" local="description"
TEXT: nodeName="#text" nodeValue="An in-depth look at creating applications
with XML."
TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="book" local="book"
ATTR: nodeName="id" local="id" nodeValue="bk102"
TEXT: nodeName="#text" nodeValue="bk102"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="author" local="author"
TEXT: nodeName="#text" nodeValue="Ralls, Kim"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="title" local="title"
TEXT: nodeName="#text" nodeValue="Midnight Rain"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="genre" local="genre"
TEXT: nodeName="#text" nodeValue="Fantasy"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="price" local="price"
TEXT: nodeName="#text" nodeValue="5.95"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="publish_date" local="publish_date"
```

PHP – SOAP - SAX

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
 sax	Pasta de ficheiros	
 books.xml	Documento XML	5 KB
 nusoap.php	Ficheiro PHP	347 KB
 pag_clienteSOAP.php	Ficheiro PHP	1 KB
 servicoSOAP.php	Ficheiro PHP	1 KB
 simple.xml	Documento XML	2 KB

servicoSOAP.php – ficheiro que trata de ler o ficheiro XML com o SAX e de o enviar ao cliente num envelope SOAP

pag_ClienteSOAP.php – cliente SOAP em php que chama o servicoSOAP.php



PHP – SOAP - SAX

Parte do Código serviçoSOAP.php

```
1  ?>php
2  require_once "nusoap.php";
3  function obterDados($nomeFicheiroACorrer) {
4      // -- verifica se shell_exec está ativo
5      if(!function_exists('shell_exec')) {
6          die("A função shell_exec deve estar ativa!");
7      }
8
9      $output = shell_exec('java sax/SAXLocalNameCount ' . $nomeFicheiroACorrer);
10     $XML = "";
11
12     $XML = explode('Local Name', $output, 1);
13     $XML = substr($output, strpos($output, "Local Name"));
14
15     return $XML;
16
17 }
18
19 @var soap_server $server _emvia
20 $server = new soap_server();
21 $server-&gt;configureWSDL('AddService', 'obtemXML');
22 $server-&gt;wsdl-&gt;schemaTargetNamespace = 'obtemXML';
23 $server-&gt;register("obterDados",
24     array('tipo' =&gt; "xsd:string"),
25     array('return' =&gt; 'xsd:string')
26 );
27 $server-&gt;service(file_get_contents('php://input'));
28 ?&gt;</pre
```



PHP – SOAP - SAX

Código do pag_clienteSOAP.php

```
1 <?php
2 require_once "nusoap.php";
3
4 $client = new nusoap_client("http://localhost/IS_WebServices/TP_SAX_SOAP_sendbooks/servicoSOAP.php");
5
6
7 $error = $client->getError();
8 if ($error) {
9     echo "<h2>Constructor error</h2><pre>" . $error . "</pre>";
10 }
11
12 $result = $client->call("obterDados", array("nome" => "books.xml"));
13
14 if ($client->fault) {
15     echo "<h2>Fault</h2><pre>" . $result . "</pre>";
16 } else {
17     $error = $client->getError();
18     if ($error) {
19         echo "<h2>Error</h2><pre>" . $error . "</pre>";
20     }else {
21         header("content-type: text/plain");
22         echo $result;
23     }
24 }
25 ?>
```



PHP – SOAP - SAX

Teste SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane displays the XML request sent to the service, and the right pane displays the XML response received from the service.

Request:

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header/>
    <soapenv:Body>
        <obterDados soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
            <tipo xsi:type="xsd:string">books.xml</tipo>
        </obterDados>
    </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns1:obterDadosResponse xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/">
            <return xsi:type="xsd:string">Local Name "genre" occurs 12 times
Local Name "catalog" occurs 1 times
Local Name "book" occurs 12 times
Local Name "author" occurs 12 times
Local Name "description" occurs 12 times
Local Name "price" occurs 12 times
Local Name "publish_date" occurs 12 times
Local Name "title" occurs 12 times</return>
        </ns1:obterDadosResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```



PHP – SOAP - SAX

Teste Browser (pag_clienteSOAP.php)

```
Local Name "genre" occurs 12 times
Local Name "catalog" occurs 1 times
Local Name "book" occurs 12 times
Local Name "author" occurs 12 times
Local Name "description" occurs 12 times
Local Name "price" occurs 12 times
Local Name "publish_date" occurs 12 times
Local Name "title" occurs 12 times
```



PHP – SOAP - JDOM

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
books.xml	Documento XML	5 KB
clienteSOAP.php	Ficheiro PHP	1 KB
correr.bat	Ficheiro batch do ...	1 KB
jdom-2.0.6.jar	Executable Jar File	298 KB
jdom-2.0.6-contrib.jar	Executable Jar File	164 KB
jdom-2.0.6-javadoc.jar	Executable Jar File	984 KB
jdom-2.0.6-junit.jar	Executable Jar File	483 KB
jdom-2.0.6-sources.jar	Executable Jar File	841 KB
LICENSE.txt	Documento de te...	3 KB
nusoap.php	Ficheiro PHP	347 KB
pag_carregaFicheiro.php	Ficheiro PHP	1 KB
pag_clienteSOAP.php	Ficheiro PHP	1 KB
PlayWithJDom.class	Ficheiro CLASS	2 KB
PlayWithJDom.java	Ficheiro JAVA	2 KB
README.txt	Documento de te...	5 KB
servicoSOAP.php	Ficheiro PHP	2 KB
simple.xml	Documento XML	2 KB
upload.php	Ficheiro PHP	2 KB
VARIAVEIS AMBIENTE A DEFINIR.txt	Documento de te...	1 KB

servicoSOAP.php – ficheiro que trata de ler o ficheiro XML com o JDOM e de o enviar ao cliente num envelope SOAP

pag_ClienteSOAP.php – cliente SOAP em php que chama o servicoSOAP.php



PHP – SOAP - JDOM

Parte do Código serviçoSOAP.php

```
1  ?>php
2  require_once "nusoap.php";
3  function obterDados($nomeFicheiroACorrer) {
4      // -- verifica se shell_exec está ativo
5      if(function_exists('shell_exec')) {
6          die("A função shell_exec deve estar ativa!");
7      }
8
9      // -- gera uma string aleatória para colocar no nome do ficheiro bat a correr, de modo a evitar sobreposição se o web service for chamado mais que uma vez ao mesmo tempo
10     function generateRandomString($length = 10) {
11         return substr(str_shuffle(str_repeat($x='0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ', ceil($length/strlen($x)) )),1,$length);
12     }
13
14     $nomeFicheiro = generateRandomString(15) . ".bat";
15
16
17     // -- gerar .bat para correr o JDOM
18     // -- -- necessário devido à necessidade de definir várias variáveis PATH e CLASSPATH
19     $textoFicheiro = 'SET PATH=%PATH%;C:\testar\orpc.dev\bin\win32
20     SET PATH=%PATH%;C:\Program Files\Java\jdk1.8.0_221\bin
21     SET CLASSPATH=%CLASSPATH%;C:\testar\orpc.dev\lib\orpc.jar;C:\testar\jdom\jdom-2.0.6.jar;
22     java PlayWithJDom " . $nomeFicheiroACorrer;
23     // -- coloca conteúdo no correr.bat
24     file_put_contents($nomeFicheiro, $textoFicheiro);
25
26
27     // -- corre o ficheiro .bat gerado.
28     $output = shell_exec($nomeFicheiro);
29     $XML = explode('&lt;?xml version="1.0" encoding="UTF-8"?&gt;', $output)[1];
30
31     // -- Remove ficheiro gerado
32     unlink($nomeFicheiro);
33
34     return $XML;
35
36 }
37
38 // -- cria servidor SOAP e envia
39 $server = new soap_server();
40 $server-&gt;configureWSL('AddService', 'obtemXML');
41 $server-&gt;wsdl-&gt;schemaTargetNamespace = 'obtemXML';
42 $server-&gt;register("obterDados",
43     array('tipo' =&gt; "xsd:string"),
44     array('return' =&gt; 'xsd:string')
45 );
46 $server-&gt;service(file_get_contents('php://input'));
47 ?&gt;</pre
```



PHP – SOAP - JDOM

Código do pag_clienteSOAP.php

```
1  ?>php
2  require_once "nusoap.php";
3
4  $client = new nusoap_client("http://localhost/IS_WebServices/TP_JDOM_SOAP_sendbooks/servicoSOAP.php");
5
6
7  $error = $client-&gt;getError();
8  if ($error) {
9      echo "&lt;h2&gt;Constructor error&lt;/h2&gt;&lt;pre&gt;" . $error . "&lt;/pre&gt;";
10 }
11
12 $result = $client-&gt;call("obterDados", array("nome" =&gt; "books.xml"));
13
14 if ($client-&gt;fault) {
15     echo "&lt;h2&gt;Fault&lt;/h2&gt;&lt;pre&gt;" . $result . "&lt;/pre&gt;";
16 }
17 else {
18     $error = $client-&gt;getError();
19     if ($error) {
20         echo "&lt;h2&gt;Error&lt;/h2&gt;&lt;pre&gt;" . $error . "&lt;/pre&gt;";
21     }
22     else {
23         header("content-type: text/xml");
24         echo $result;
25     }
26 }
27
28
29 ?&gt;</pre
```



PHP – SOAP - JDOM

Teste SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane displays a SOAP request message, and the right pane displays the XML response message.

Request:

```
<?xml version="1.0"?>
<soapenv:Envelope xmlns:ns1="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header></soapenv:Header>
    <soapenv:Body>
        <obterDados soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
            <tipo xsi:type="xsd:string">books.xml</tipo>
        </obterDados>
    </soapenv:Body>
</soapenv:Envelope>
```

Response:

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <SOAP-ENV:Header></SOAP-ENV:Header>
    <ns1:obterDadosResponse xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/">
        <return xsi:type="xsd:string"><![CDATA[<catalog>
<book id="bk101">
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications with XML.</description>
</book>
<book id="bk102">
    <author>Halli, Kim</author>
    <title>Midnight Rain</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-12-16</publish_date>
    <description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
</book>
<book id="bk103">
    <author>Corets, Eva</author>
    <title>Maeve Ascendant</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2000-11-17</publish_date>
    <description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description>
</book>
<book id="bk104">
    <author>Corets, Eva</author>
    <title>Oberon's Legacy</title>
    <genre>Fantasy</genre>
    <price>5.95</price>
    <publish_date>2001-03-10</publish_date>
    <description>In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Samuel ro Maues
```

At the bottom of the SoapUI interface, there are tabs for Auth, Headers (0), Attachments (0), WS-A, WS-RM, JMS Headers, JMS Property (0) on the left, and Headers (10), Attachments (0), SSL Info, WSS (0), JMS (0) on the right.



PHP – SOAP - JDOM

Teste Browser (pag_clienteSOAP.php)

The screenshot shows a Java IDE interface with an XML editor and an outline view.

XML Editor:

```
<catalog>
  <book id="bk101">
    <author> Gambardella, Matthew </author>
    <title> XML Developer's Guide </title>
    <genre> Computer </genre>
    <price> 44.95 </price>
    <publish_date> 2000-10-01 </publish_date>
    <description> An in-depth look at creating applications with XML. </description>
  </book>
  <book id="bk102">
    <author> Reilly, Kim </author>
    <title> Midnight Rain </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-12-16 </publish_date>
    <description>
      A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.
    </description>
  </book>
  <book id="bk103">
    <author> Corets, Eva </author>
    <title> Maeve Ascendant </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-11-17 </publish_date>
    <description>
      After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.
    </description>
  </book>
  <book id="bk104">
    <author> Corets, Eva </author>
    <title> Oberon's Legacy </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2001-03-10 </publish_date>
    <description>
      In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Maeve Ascendant.
    </description>
  </book>
  <book id="bk105">
    <author> Corets, Eva </author>
```

Outline View:

- catalog
 - book id: bk101
 - author
 - title
 - genre
 - price
 - publish_date
 - description
 - book id: bk102
 - author
 - title
 - genre
 - price
 - publish_date
 - description
 - book id: bk103
 - author
 - title
 - genre
 - price
 - publish_date
 - description
 - book id: bk104
 - author
 - title
 - genre
 - price
 - publish_date
 - description
 - book id: bk105
 - author
 - title
 - genre
 - price
 - publish_date
 - description



PHP – SOAP - JAXP

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
stax	Pasta de ficheiros	
books.xml	Documento XML	5 KB
nusoap.php	Ficheiro PHP	347 KB
pag_clienteSOAP.php	Ficheiro PHP	1 KB
servicoSOAP.php	Ficheiro PHP	1 KB
simple.xml	Documento XML	2 KB
VARIAVEIS AMBIENTE A DEFINIR.txt	Documento de te...	1 KB

servicoSOAP.php – ficheiro que trata de ler o ficheiro XML com o JAXP e de o enviar ao cliente num envelope SOAP

pag_ClienteSOAP.php – cliente SOAP em php que chama o servicoSOAP.php



PHP – SOAP - JAXP

Parte do Código serviçoSOAP.php

```
1  <?php
2  require_once "nusoap.php";
3  function obterDados($nomeFicheiroACorrer) {
4      // -- verifica se shell_exec está ativo
5      if(!function_exists('shell_exec')) {
6          die("A função shell_exec deve estar ativa!");
7      }
8
9      $output = shell_exec('java stax/cursor/CursorParse ' . $nomeFicheiroACorrer);
10     $XML = "";
11
12     $XML = explode('<?xml version="1.0" ', $output)[1];
13     $XML = explode('?>', $XML)[1];
14
15     $XML = explode(' Parsing Time = ', $XML)[0];
16
17     return $XML;
18
19 }
20
21 // -- cria servidor SOAP e envia
22 $server = new soap_server();
23 $server->configureWSDL('AddService', 'obtemXML');
24 $server->wsdl->schemaTargetNamespace = 'obtemXML';
25 $server->register("obterDados",
26     array('tipo' => 'xsd:string'),
27     array('return' => 'xsd:string')
28 );
29 $server->service(file_get_contents('php://input'));
30 ?>
```



PHP – SOAP - JAXP

Código do pag_clienteSOAP.php

```
1  <?php
2  require_once "nusoap.php";
3
4  $client = new nusoap_client("http://localhost/IS_WebServices/TP_JAXP_SOAP_sendbooks/servicoSOAP.php");
5
6
7  $error = $client->getError();
8  if ($error) {
9      echo "<h2>Constructor error</h2><pre>" . $error . "</pre>";
10 }
11
12 $result = $client->call("obterDados", array("nome" => "books.xml"));
13
14 if ($client->fault) {
15     echo "<h2>Fault</h2><pre>" . $result . "</pre>";
16 } else {
17     $error = $client->getError();
18     if ($error) {
19         echo "<h2>Error</h2><pre>" . $error . "</pre>";
20     } else {
21         header("content-type: text/xml");
22         echo $result;
23     }
24 }
25 }
```



PHP – SOAP - JAXP

Teste SoapUI

The screenshot shows the SoapUI interface with a successful XML response. The left pane displays the XML structure of the request, and the right pane shows the detailed XML response. The response contains four book entries, each with an ID, author, title, genre, price, publish date, and a description.

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Header></soapenv:Header>
<soapenv:Body>
<ns1:obterDados soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<tipo xsi:type="xsd:string">books.xml</tipo>
</ns1:obterDados>
</soapenv:Body>
</soapenv:Envelope>
```

```
SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
<ns1:obterDadosResponse xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/">
<return xsi:type="xsd:string"><![CDATA[<catalog>
<book id="bk101">
<author>Gambardella, Matthew</author>
<title>XML Developer's Guide</title>
<genre>Computer</genre>
<price>44.95</price>
<publish_date>2000-10-01</publish_date>
<description>An in-depth look at creating applications with XML.</description>
</book>
<book id="bk102">
<author>Ralls, Kim</author>
<title>Midnight Rain</title>
<genre>Fantasy</genre>
<price>5.95</price>
<publish_date>2000-12-16</publish_date>
<description>A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.</description>
</book>
<book id="bk103">
<author>Corets, Eva</author>
<title>Mave Ascendant</title>
<genre>Fantasy</genre>
<price>5.95</price>
<publish_date>2000-11-17</publish_date>
<description>After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.</description>
</book>
<book id="bk104">
<author>Corets, Eva</author>
<title>Oberon's Legacy</title>
<genre>Fantasy</genre>
<price>5.95</price>
<publish_date>2001-03-10</publish_date>
<description>In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Samuel no Mau&gt;
```

Auth Headers (0) Attachments (0) WS-A WS-RM JMS Headers JMS Property (0)
Headers (10) Attachments (0) SSL Info WSS (0) JMS (0)

response time: 164ms (6122 bytes)



PHP – SOAP - JAXP

Teste Browser (pag_clienteSOAP.php)

```
Search by name or XPath

<catalog>
  <book id="bk101">
    <author> Gambarella, Matthew </author>
    <title> XML Developer's Guide </title>
    <genre> Computer </genre>
    <price> 44.95 </price>
    <publish_date> 2000-10-01 </publish_date>
    <description> An in-depth look at creating applications with XML. </description>
  </book>
  <book id="bk102">
    <author> Ralls, Kim </author>
    <title> Midnight Rain </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-12-16 </publish_date>
    <description>
      A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.
    </description>
  </book>
  <book id="bk103">
    <author> Corets, Eva </author>
    <title> Maeve Ascendant </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-11-17 </publish_date>
    <description>
      After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.
    </description>
  </book>
  <book id="bk104">
    <author> Corets, Eva </author>
    <title> Oberon's Legacy </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2001-03-10 </publish_date>
    <description>
      In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Maeve Ascendant.
    </description>
  </book>
  <book id="bk105">
```

PHP – SOAP - JAXB

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
generated	Pasta de ficheiros	
Main.class	Ficheiro CLASS	3 KB
nusoap.php	Ficheiro PHP	347 KB
pag_clienteSOAP.php	Ficheiro PHP	1 KB
po.xml	Documento XML	3 KB
run_testes.bat	Ficheiro batch do ...	1 KB
servicoSOAP.php	Ficheiro PHP	1 KB
VARIAVEIS AMBIENTE A DEFINIR.txt	Documento de te...	1 KB

servicoSOAP.php – ficheiro que trata de ler o ficheiro XML com o JAXB e de o enviar ao cliente num envelope SOAP

pag_ClienteSOAP.php – cliente SOAP em php que chama o servicoSOAP.php



PHP – SOAP - JAXB

Parte do Código serviçoSOAP.php

```
IP_JAXB_SOAP_sendpo > servicioSOAP.php > ...
1  |?php
2  |require_once "nusoap.php";
3  |function obterDados($nomeFicheiroACorrer) {
4  |    // -- verifica se shell_exec está ativo
5  |    if(!function_exists('shell_exec')) {
6  |        die("A função shell_exec deve estar ativa!");
7  |
8  |
9  |    $output = shell_exec('java Main ' . $nomeFicheiroACorrer);
10 |    $XML = $output;
11 |    return $XML;
12 |
13 }
14
15 // -- cria servidor SOAP e envia
16 $server = new soap_server();
17 $server->configureWSDL('AddService', 'obtemXML');
18 $server->wsdl->schemaTargetNamespace = 'obtemXML';
19 $server->register("obterDados",
20     array('tipo' => "xsd:string"),
21     array('return' => 'xsd:string')
22 );
23 $server->service(file_get_contents('php://input'));
24 ?>
```

PHP – SOAP - JAXB

Código do pag_clienteSOAP.php

```
1  <?php
2  require_once "nusoap.php";
3
4  $client = new nusoap_client("http://localhost/IS_WebServices/TP_JAXB_SOAP_sendpo/servicoSOAP.php");
5
6
7  $error = $client->getError();
8  if ($error) {
9      echo "<h2>Constructor error</h2><pre>" . $error . "</pre>";
10 }
11
12 $result = $client->call("obterDados", array("nome" => "po.xml"));
13
14 if ($client->fault) {
15     echo "<h2>Fault</h2><pre>" . $result . "</pre>";
16 } else {
17     $error = $client->getError();
18     if ($error) {
19         echo "<h2>Error</h2><pre>" . $error . "</pre>";
20     } else {
21         header("content-type: text/plain");
22         echo $result;
23     }
24 }
25 }
```



PHP – SOAP - JAXB

Teste SoapUI

The screenshot shows the SoapUI interface with two panes. The left pane displays the XML request sent to the service, and the right pane displays the XML response received from the service.

Request (Left Pane):

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://schemas.xmlsoap.org/soap/encoding/">
  <soapenv:Header/>
  <soapenv:Body>
    <obterDados soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <tipo xsi:type="xsd:string">po.xml</tipo>
    </obterDados>
  </soapenv:Body>
</soapenv:Envelope>
```

Response (Right Pane):

```
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Header/>
  <ns1:obterDadosResponse xmlns:ns1="http://schemas.xmlsoap.org/soap/envelope/">
    <return xsi:type="xsd:string">Ship the following items to:  

      Alice Smith  

      123 Maple Street  

      Cambridge, MA 12345  

      US  

      6 copies of "Nosferatu - Special Edition (1929)"  

      3 copies of "The Mummy (1965)"  

      3 copies of "Godzilla and Mothra: Battle for Earth/Godzilla vs. King Ghidora"<!--&lt;br/&gt;--&gt;
    &lt;/ns1:obterDadosResponse&gt;
  &lt;/SOAP-ENV:Header&gt;
&lt;/SOAP-ENV:Envelope&gt;</pre>

At the bottom of the SoapUI interface, there are tabs for Headers (9), Attachments (0), SSL Info, WSS (0), and JMS (0).


```



PHP – SOAP - JAXB

Teste Browser (pag_clienteSOAP.php)

Ship the following items to:

Alice Smith
123 Maple Street
Cambridge, MA 12345
US

5 copies of "Nosferatu - Special Edition (1929)"
3 copies of "The Mummy (1959)"
3 copies of "Godzilla and Mothra: Battle for Earth/Godzilla vs. King Ghidora"



PHP – REST - DOM

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
dom	Pasta de ficheiros	
sax	Pasta de ficheiros	
books.xml	Documento XML	5 KB
pag_clienteREST.php	Ficheiro PHP	1 KB
php_chamaAPI.php	Ficheiro PHP	1 KB
servicoREST.php	Ficheiro PHP	1 KB
simple.xml	Documento XML	2 KB

servicoREST.php – ficheiro que trata de ler o ficheiro XML com o DOM e de o enviar ao client

pag_ClienteREST.php – cliente REST em php que chama o servicoREST.php através do php_chamaAPI.php.



PHP – REST - DOM

Parte do Código serviçoREST.php

```
IP_DOM_REST_sendbooks > servicioREST.php > ...
1  <?php
2  function obterDados($nomeFicheiroACorrer) {
3      // -- verifica se shell_exec está ativo
4      if(!function_exists('shell_exec')) {
5          die("A função shell_exec deve estar ativa!");
6      }
7
8      $output = shell_exec('java dom/DOMEcho ' . $nomeFicheiroACorrer);
9      $XML = "";
10
11     $XML = "DOC: nodeName=" . explode('DOC: nodeName=', $output)[1];
12     $XML = explode(' Parsing Time = ', $XML)[0];
13
14     return $XML;
15
16 }
17
18 // -- envia dados
19
20 // --- nome de ficheiro enviado por get
21 if(!empty($_GET['nomeFicheiroXML'])){
22     echo obterDados($_GET['nomeFicheiroXML']);
23 } else {
24     echo "Ocorreu um erro -- não foi dado um nome de ficheiro";
25 }
26
27
28 ?>
```



PHP – REST - DOM

Código do pag_clienteREST.php

```
1 <?php
2     require("php_chamaAPI.php");
3     $url = "http://localhost/IS_WebServices/TP_DOM_REST_sendbooks/servicoREST.php";
4     $nomeFicheiro = "books.xml";
5
6     $make_call = callAPI('GET', $url . "?nomeFicheiroXML=" . urldecode($nomeFicheiro), null);
7     $response = $make_call;
8
9     header("Content-Type: text/plain");
10    echo $response;
11 ?>
```



PHP – REST - DOM

Teste POSTMAN

The screenshot shows the POSTMAN interface with the following details:

- Request Type:** GET
- URL:** http://localhost/IS_WebServices/TP_DOM_REST_sendbooks/servicoreST.php?nomeFicheiroXML=books.xml
- Params Tab:** The "Params" tab is selected, showing a single parameter: nomeFicheiroXML with a value of books.xml.
- Body Tab:** The "Pretty" option is selected, displaying the XML response in a readable format. The XML content is as follows:

```
1 DOC: nodeName="#document"
2 ELEM: nodeName="catalog" local="catalog"
3 TEXT: nodeName="#text" nodeValue=[W5]
4 ELEM: nodeName="book" local="book"
5 ATTR: nodeName="id" local="id" nodeValue="bk101"
6 TEXT: nodeName="#text" nodeValue="bk101"
7 TEXT: nodeName="#text" nodeValue=[W5]
8 ELEM: nodeName="author" local="author"
9 TEXT: nodeName="#text" nodeValue="Gambardella, Matthew"
10 TEXT: nodeName="#text" nodeValue=[W5]
11 ELEM: nodeName="title" local="title"
12 TEXT: nodeName="#text" nodeValue="XML Developer's Guide"
13 TEXT: nodeName="#text" nodeValue=[W5]
14 ELEM: nodeName="genre" local="genre"
15 TEXT: nodeName="#text" nodeValue="Computer"
16 TEXT: nodeName="#text" nodeValue=[W5]
17 ELEM: nodeName="price" local="price"
18 TEXT: nodeName="#text" nodeValue="44,95"
19 TEXT: nodeName="#text" nodeValue=[W5]
20 ELEM: nodeName="publish_date" local="publish_date"
21 TEXT: nodeName="#text" nodeValue="2000-10-01"
22 TEXT: nodeName="#text" nodeValue=[W5]
23 ELEM: nodeName="description" local="description"
24 TEXT: nodeName="#text" nodeValue="An in-depth look at creating applications
with XML."
25 TEXT: nodeName="#text" nodeValue=[W5]
26 TEXT: nodeName="#text" nodeValue=[W5]
27 ELEM: nodeName="book" local="book"
28 ATTR: nodeName="id" local="id" nodeValue="bk102"
29 TEXT: nodeName="#text" nodeValue="bk102"
30 TEXT: nodeName="#text" nodeValue=[W5]
31 ELEM: nodeName="author" local="author"
32 FLEM: nodeName="author" local="author"
```

Status: 200 OK | Time: 185ms | Size: 14.65 KB | Save Response ▾



PHP – REST - DOM

Teste Browser (pag_clienteREST.php)

```
DOC: nodeName="#document"
ELEM: nodeName="catalog" local="catalog"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="book" local="book"
ATTR: nodeName="#id" local="id" nodeValue="bk101"
TEXT: nodeName="#text" nodeValue="bk101"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="author" local="author"
TEXT: nodeName="#text" nodeValue="Gambardella, Matthew"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="title" local="title"
TEXT: nodeName="#text" nodeValue="XML Developer's Guide"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="genre" local="genre"
TEXT: nodeName="#text" nodeValue="Computer"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="price" local="price"
TEXT: nodeName="#text" nodeValue="44.95"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="publish_date" local="publish_date"
TEXT: nodeName="#text" nodeValue="2000-10-01"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="description" local="description"
TEXT: nodeName="#text" nodeValue="An in-depth look at creating applications
with XML."
TEXT: nodeName="#text" nodeValue=[WS]
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="book" local="book"
ATTR: nodeName="#id" local="id" nodeValue="bk102"
TEXT: nodeName="#text" nodeValue="bk102"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="author" local="author"
TEXT: nodeName="#text" nodeValue="Ralls, Kim"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="title" local="title"
TEXT: nodeName="#text" nodeValue="Midnight Rain"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="genre" local="genre"
TEXT: nodeName="#text" nodeValue="Fantasy"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="price" local="price"
TEXT: nodeName="#text" nodeValue="5.95"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="publish_date" local="publish_date"
TEXT: nodeName="#text" nodeValue="2000-12-16"
TEXT: nodeName="#text" nodeValue=[WS]
ELEM: nodeName="description" local="description"
TEXT: nodeName="#text" nodeValue=[WS]
```



PHP – REST - SAX

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
sax	Pasta de ficheiros	
books.xml	Documento XML	5 KB
pag_clienteREST.php	Ficheiro PHP	1 KB
php_chamaAPI.php	Ficheiro PHP	1 KB
servicoREST.php	Ficheiro PHP	1 KB
simple.xml	Documento XML	2 KB

`servicoREST.php` – ficheiro que trata de ler o ficheiro XML com o SAX e de enviar o resultado ao cliente

`pag_ClienteREST.php` – cliente REST em php que chama o `servicoREST.php` através do `php_chamaAPI.php`.



PHP – REST - SAX

Parte do Código serviçoREST.php

```
TP_SAX_REST_sendbooks > servicioREST.php > ...
1  ?>php
2  function obterDados($nomeFicheiroACorrer) {
3      // -- verifica se shell_exec está ativo
4      if(!function_exists('shell_exec')) {
5          die("A função shell_exec deve estar ativa!");
6      }
7
8      $output = shell_exec('java sax/SAXLocalNameCount ' . $nomeFicheiroACorrer);
9      $XML = "";
10
11     $XML = explode('Local Name', $output, 1);
12     $XML = substr($output, strpos($output, "Local Name"));
13
14     return $XML;
15 }
16
17 // -- envia dados
18
19 // -- nome de ficheiro enviado por get
20 if(!empty($_GET['nomeFicheiroXML'])){
21     echo obterDados($_GET['nomeFicheiroXML']);
22 } else {
23     echo "Ocorreu um erro -- não foi dado um nome de ficheiro";
24 }
25 ?php</pre
```



PHP – REST - SAX

Código do pag_clienteREST.php

```
TP_SAX_REST_sendbooks > pag_clienteREST.php > ...
1 <?php
2     require("php_chamaAPI.php");
3     $url = "http://localhost/IS_WebServices/TP_SAX_REST_sendbooks/servicoREST.php";
4     $nomeFicheiro = "books.xml";
5
6     $make_call = callAPI('GET', $url . "?nomeFicheiroXML=" . urldecode($nomeFicheiro), null);
7     $response = $make_call;
8
9     header("Content-Type: text/plain");
10    echo $response;
11 ?>
```



PHP – REST - SAX

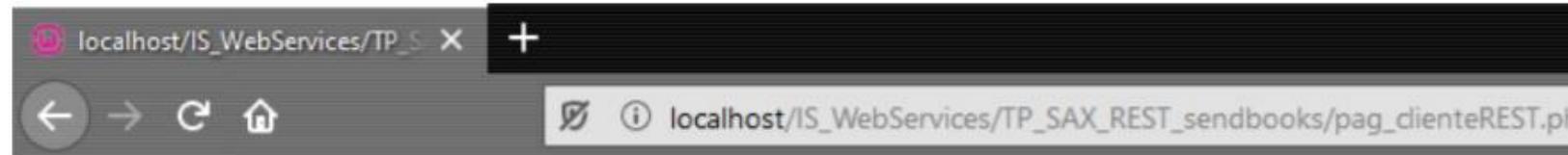
Teste POSTMAN

The screenshot shows the POSTMAN interface with the following details:

- Request URL:** GET http://localhost/I5_WebServices/TP_SAX_REST_sendbooks/servicoREST.php?nomeficheiroXML=books.xml
- Params Tab:** Shows a single parameter named "nomeficheiroXML" with the value "books.xml".
- Body Tab:** Displays the XML response from the server, which is a catalog containing 12 books. The XML structure includes elements like "genre", "catalog", "book", "author", "description", "price", "publish_date", and "title".
- Status Bar:** Status: 200 OK, Time: 508ms, Size: 535.6, Save Response.

PHP – REST - SAX

Teste Browser (pag_clienteREST.php)



```
Local Name "genre" occurs 12 times
Local Name "catalog" occurs 1 times
Local Name "book" occurs 12 times
Local Name "author" occurs 12 times
Local Name "description" occurs 12 times
Local Name "price" occurs 12 times
Local Name "publish_date" occurs 12 times
Local Name "title" occurs 12 times
```

PHP – REST - JDOM

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
books.xml	Documento XML	5 KB
jdom-2.0.6.jar	Executable Jar File	298 KB
jdom-2.0.6-contrib.jar	Executable Jar File	164 KB
jdom-2.0.6-javadoc.jar	Executable Jar File	984 KB
jdom-2.0.6-junit.jar	Executable Jar File	483 KB
jdom-2.0.6-sourcesjar	Executable Jar File	841 KB
LICENSE.txt	Documento de te...	3 KB
pag_clienteREST.php	Ficheiro PHP	1 KB
php_chamaAPI.php	Ficheiro PHP	1 KB
PlayWithJDom.class	Ficheiro CLASS	2 KB
PlayWithJDom.java	Ficheiro JAVA	2 KB
README.txt	Documento de te...	5 KB
servicoREST.php	Ficheiro PHP	2 KB
simple.xml	Documento XML	2 KB

servicoREST.php – ficheiro que trata de ler o ficheiro XML com o JDOM e de o enviar ao client

pag_ClienteREST.php – cliente REST em php que chama o servicoREST.php através do **php_chamaAPI.php**.



PHP – REST - JDOM

Parte do Código serviçoREST.php

```
cd /home/nestor/verbaedita > rm serviceREST.php > ...  
1  ?php  
2  function obterDados($nomeFicheiroACorrer) {  
3      // -- verifica se shell_exec está ativo  
4      if(function_exists('shell_exec')) {  
5          die("A função shell_exec deve estar ativa");  
6      }  
7  
8      // -- gera uma string aleatória para colocar no nome do ficheiro bat a correr, de modo a evitar sobreposição se o web service for chamado mais que uma vez ao mesmo tempo  
9      function generateRandomString($length = 10) {  
10          return substr(str_shuffle(str_repeat($x='0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ', ceil($length/strlen($x)) )),1,$length);  
11      }  
12  
13      $nomeFicheiro = generateRandomString(15) . ".bat";  
14  
15  
16      // -- gerar .bat para correr o JDOM  
17      // --- necessário devido à necessidade de definir várias variáveis PATH e CLASSPATH  
18      $textoFicheiro = 'SET PATH=%PATH%;C:\testar\orpc.dev\bin\win32  
19      SET PATH=%PATH%;C:\Program Files\Java\jdk1.8.0_221\bin"  
20      SET CLASSPATH=%CLASSPATH%;C:\testar\orpc.dev\lib\orpc.jar;C:\testar\jdom\jdom-2.0.6.jar;  
21      java PlayWithJDom '$nomeFicheiroCorrer';  
22      // -- coloca conteúdo no correr.bat  
23      file_put_contents($nomeFicheiro, $textoFicheiro);  
24  
25  
26      // -- corre o ficheiro .bat gerado.  
27      $output = shell_exec($nomeFicheiro);  
28      $XML = explode('<?xml version="1.0" encoding="UTF-8"?>', $output)[1];  
29  
30      // -- remove ficheiro gerado  
31      unlink($nomeFicheiro);  
32  
33      return $XML;  
34  }  
35  
36  // -- envia dados  
37  
38  // -- nome de ficheiro enviado por get  
39  if(!empty($_GET['nomeFicheiroXML'])){  
40      echo obterDados($_GET['nomeFicheiroXML']);  
41  } else {  
42      echo "Ocorreu um erro -- não foi dado um nome de ficheiro";  
43  }  
44  }  
45  ?>
```

① Do you want to exclude the Visual Studio
settings files (.classpath, .project, .settings)
from the file explorer?



PHP – REST - JDOM

Código do pag_clienteREST.php

```
TP_JDOM_REST_sendbooks > pag_clienteREST.php > ...
1 <?php
2     require("php_chamaAPI.php");
3     $url = "http://localhost/IS_WebServices/TP_JDOM_REST_sendbooks/servicoREST.php";
4     $nomeFicheiro = "books.xml";
5
6     $make_call = callAPI('GET', $url . "?nomeFicheiroXML=" . urldecode($nomeFicheiro), null);
7     $response = $make_call;
8
9     header("Content-Type: text/xml");
10    echo $response;
11 ?>
```



PHP – REST - JDOM

Teste POSTMAN

The screenshot shows the POSTMAN interface with the following details:

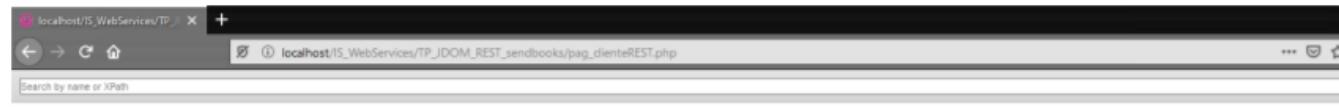
- Method:** GET
- URL:** http://localhost/IS_WebServices/TP_JDOM_REST_sendbooks/servicoREST.php?nomeFicheiroXML=books.xml
- Params:** A table showing a single parameter: nomeFicheiroXML with value books.xml.
- Body:** The response body displays the XML content of the 'books.xml' file, which lists four books with their details like title, author, price, and description.

```
1 <catalog>
2   <book id="bk101">
3     <author>Gambardella, Matthew</author>
4     <title>XML Developer's Guide</title>
5     <genre>Computer</genre>
6     <price>44.95</price>
7     <publish_date>2000-10-01</publish_date>
8     <description>An in-depth look at creating applications
9       with XML.</description>
10    </book>
11    <book id="bk102">
12      <author>Ralls, Kim</author>
13      <title>Midnight Rain</title>
14      <genre>Fantasy</genre>
15      <price>5.95</price>
16      <publish_date>2000-12-16</publish_date>
17      <description>A former architect battles corporate zombies,
18      an evil sorceress, and her own childhood to become queen
19      of the world.</description>
20    </book>
21    <book id="bk103">
22      <author>Corets, Eva</author>
23      <title>Maeve Ascendant</title>
24      <genre>Fantasy</genre>
25      <price>5.95</price>
26      <publish_date>2000-11-17</publish_date>
27      <description>After the collapse of a nanotechnology
28      society in England, the young survivors lay the
29      foundation for a new society.</description>
30    </book>
31    <book id="bk104">
32      <author>Forester, John</author>
```



PHP – REST - JDOM

Teste Browser (pag_RESTphp)



A screenshot of a web browser window displaying an XML document. The URL in the address bar is `localhost/IS_WebServices/TP_JDOM_REST_sendbooks/pag_clienteREST.php`. The page content is an XML catalog of books:

```
<catalog>
  <book id="B4101">
    <author> Damorella, Matthew </author>
    <title> XML Developer's Guide </title>
    <genre> Computer </genre>
    <price> 44.95 </price>
    <publish_date> 2000-10-01 </publish_date>
    <description> An in-depth look at creating applications with XML. </description>
  </book>
  <book id="B4102">
    <author> Hall, Kim </author>
    <title> Midnight Rain </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-12-16 </publish_date>
    <description> A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world. </description>
  </book>
  <book id="B4103">
    <author> Corets, Eva </author>
    <title> Maeve Ascendant </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-11-17 </publish_date>
    <description> After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society. </description>
  </book>
  <book id="B4104">
    <author> Corets, Eva </author>
    <title> Oberon's Legacy </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2001-03-10 </publish_date>
    <description> In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Maeve Ascendant. </description>
  </book>
  <book id="B4105">
    <author> Corets, Eva </author>
    <title> The Sundered Grail </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2001-09-10 </publish_date>
    <description> The two daughters of Maeve, half-sisters, battle one another for control of England. Sequel to Oberon's Legacy. </description>
  </book>
  <book id="B4106">
    <author> Randell, Cynthia </author>
    <title> Lover Birds </title>
    <genre> Romance </genre>
  </book>
</catalog>
```



PHP – REST - JAXP

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
stax	Pasta de ficheiros	
books.xml	Documento XML	5 KB
pag_clienteREST.php	Ficheiro PHP	1 KB
php_chamaAPI.php	Ficheiro PHP	1 KB
servicoREST.php	Ficheiro PHP	1 KB
simple.xml	Documento XML	2 KB

servicoREST.php – ficheiro que trata de ler o ficheiro XML com o JAXP e de o enviar ao client

pag_ClienteREST.php – cliente REST em php que chama o servicoREST.php através do php_chamaAPI.php.



PHP – REST - JAXP

Parte do Código serviçoREST.php

```
TP_JAXP_REST_sendbooks > servicoREST.php > ...
1  ?>php
2  function obterDados($nomeFicheiroACorrer) {
3      // -- verifica se shell_exec está ativo
4      if(!function_exists('shell_exec')) {
5          die("A função shell_exec deve estar ativa!");
6      }
7
8      $output = shell_exec('java stax/cursor/CursorParse ' . $nomeFicheiroACorrer);
9      $XML = "";
10
11     $XML = explode('&lt;?xml version="1.0" ', $output)[1];
12     $XML = explode('?', $XML)[1];
13
14     $XML = explode(' Parsing Time = ', $XML)[0];
15
16     return $XML;
17 }
18
19 // -- envia dados
20
21 // -- nome de ficheiro enviado por get
22 if(!empty($_GET['nomeFicheiroXML'])){
23     echo obterDados($_GET['nomeFicheiroXML']);
24 } else {
25     echo "Ocorreu um erro -- não foi dado um nome de ficheiro";
26 }
27
28 ?php</pre
```



PHP – REST - JAXP

Código do pag_clienteREST.php

```
TP_JAXP_REST_sendbooks > pag_clienteREST.php > ...
1 <?php
2     require("php_chamaAPI.php");
3     $url = "http://localhost/IS_WebServices/TP_JAXP_REST_sendbooks/servicoREST.php";
4     $nomeFicheiro = "books.xml";
5
6     $make_call = callAPI('GET', $url . "?nomeFicheiroXML=" . urldecode($nomeFicheiro), null);
7     $response = $make_call;
8
9     header("Content-Type: text/xml");
10    echo $response;
11 ?>
```



PHP – REST - JAXP

Teste POSTMAN

The screenshot shows the POSTMAN interface with the following details:

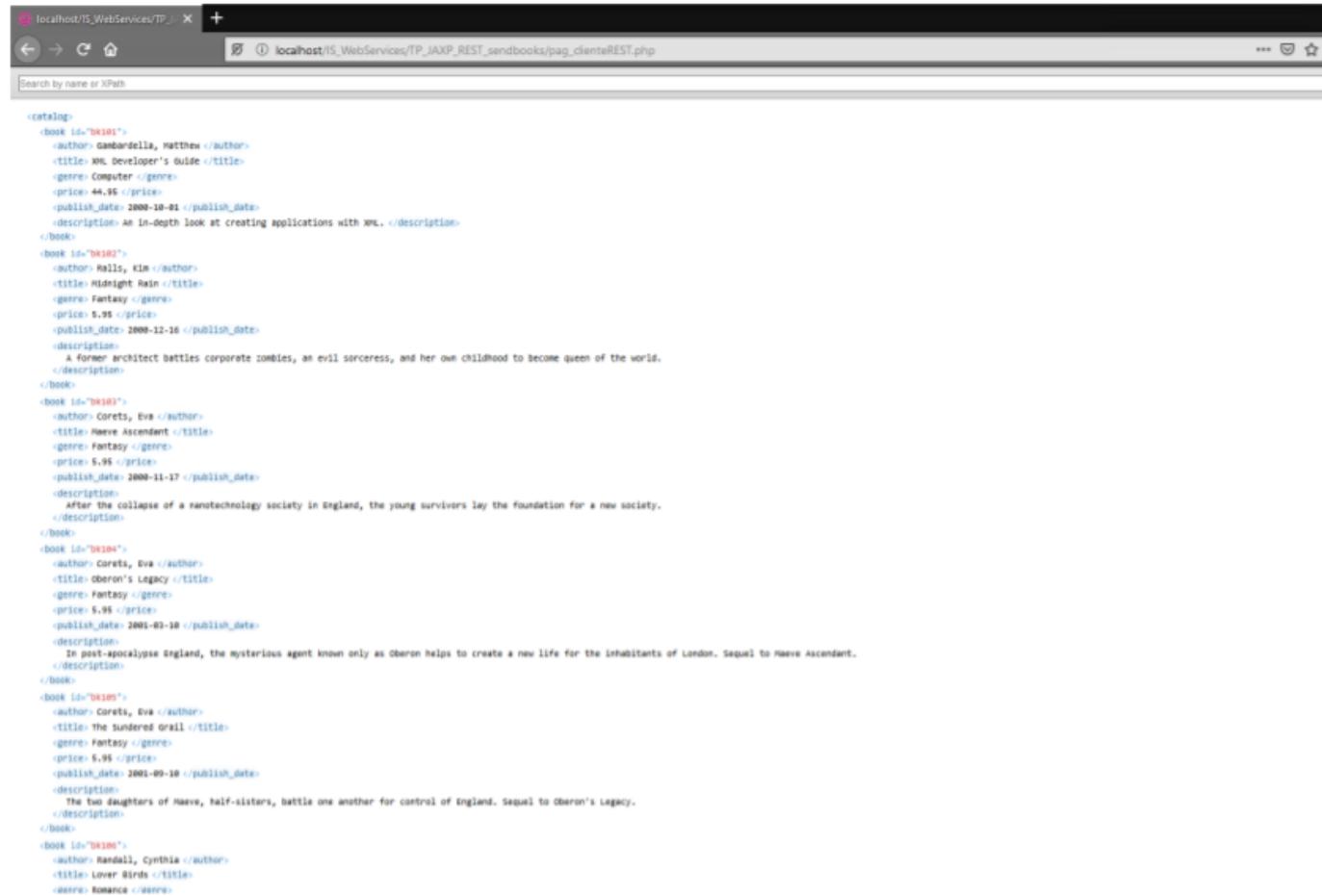
- Request Type:** GET
- URL:** http://localhost/IS_WebServices/TP_JAXP_REST_sendbooks/servicoREST.php?nomeFicheiroXML=books.xml
- Params Tab:** The "Params" tab is selected, showing a single parameter: nomeFicheiroXML with value books.xml.
- Body Tab:** The "Pretty" option is selected, displaying the XML response in a readable format.
- Response Body (Pretty):**

```
1 <catalog>
2   <book id="bk101">
3     <author>Gambardea, Matthew</author>
4     <title>XML Developer's Guide</title>
5     <genre>Computer</genre>
6     <price>44.95</price>
7     <publish_date>2000-10-01</publish_date>
8     <description>An in-depth look at creating applications
9       with XML.</description>
10    </book>
11   <book id="bk102">
12     <author>Kalls, Kim</author>
13     <title>Midnight Rain</title>
14     <genre>Fantasy</genre>
15     <price>5.95</price>
16     <publish_date>2000-12-16</publish_date>
17     <description>A former architect battles corporate zombies,
18       an evil sorceress, and her own childhood to become queen
19       of the world.</description>
20   </book>
21   <book id="bk103">
22     <author>Corets, Eva</author>
23     <title>Mewe Ascendant</title>
24     <genre>Fantasy</genre>
25     <price>5.95</price>
26     <publish_date>2000-11-17</publish_date>
27     <description>After the collapse of a nanotechnology
28       society in England, the young survivors lay the
29       foundation for a new society.</description>
30   </book>
31   <book id="bk104">
```
- Status Bar:** Status: 200 OK | Time: 46ms | Size: 4.54KB | Save Response



PHP – REST - JAXP

Teste Browser (pag_clienteREST.php)



A screenshot of a web browser window displaying an XML document. The URL in the address bar is `localhost/IS_WebServices/TP_JAXP_REST_sendbooks/pag_clienteREST.php`. The page content is an XML catalog of books:

```
<catalog>
  <book id="BK101">
    <author> Cabardella, Matthew </author>
    <title> XML Developer's Guide </title>
    <genre> Computer </genre>
    <price> 44.95 </price>
    <publish_date> 2000-10-01 </publish_date>
    <description> An in-depth look at creating applications with XML. </description>
  </book>
  <book id="BK102">
    <author> Kells, Kim </author>
    <title> Midnights Rain </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-12-16 </publish_date>
    <description> A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world. </description>
  </book>
  <book id="BK103">
    <author> Corets, Eva </author>
    <title> Heave Ascendent </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2000-11-17 </publish_date>
    <description> After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society. </description>
  </book>
  <book id="BK104">
    <author> Corets, Eva </author>
    <title> Oberon's Legacy </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2001-03-10 </publish_date>
    <description> In post-apocalypse England, the mysterious agent known only as Oberon helps to create a new life for the inhabitants of London. Sequel to Heave Ascendant. </description>
  </book>
  <book id="BK105">
    <author> Corets, Eva </author>
    <title> The Sundered Oril </title>
    <genre> Fantasy </genre>
    <price> 5.95 </price>
    <publish_date> 2001-03-10 </publish_date>
    <description> The two daughters of Heave, half-sisters, battle one another for control of England. Sequel to Oberon's Legacy. </description>
  </book>
  <book id="BK106">
    <author> Randall, Cynthia </author>
    <title> Lover Birds </title>
    <genre> Romance </genre>
  </book>
</catalog>
```



PHP – REST - JAXB

Estrutura do Projeto Criado

Nome	Tipo	Tamanho
generated	Pasta de ficheiros	
Main.class	Ficheiro CLASS	3 KB
pag_clienteREST.php	Ficheiro PHP	1 KB
php_chamaAPI.php	Ficheiro PHP	1 KB
po.xml	Documento XML	3 KB
run_testes.bat	Ficheiro batch do ...	1 KB
servicoREST.php	Ficheiro PHP	1 KB
WEB-INF\AMBIENTE\REFINERIA	Pasta de ficheiros	1 KB

servicoREST.php – ficheiro que trata de ler o ficheiro XML com o JAXB e de o enviar ao client

pag_ClienteREST.php – cliente REST em php que chama o servicoREST.php através do php_chamaAPI.php.



PHP – REST - JAXB

Parte do Código serviçoREST.php

```
TP_JAXB_REST_sendpo > servicioREST.php > ...
1  ?>php
2  function obterDados($nomeFicheiroACorrer) {
3      // -- verifica se shell_exec está ativo
4      if(!function_exists('shell_exec')) {
5          die("A função shell_exec deve estar ativa!");
6      }
7
8      $output = shell_exec('java Main ' . $nomeFicheiroACorrer);
9      $XML = $output;
10     return $XML;
11 }
12
13 // -- envia dados
14
15 // -- nome de ficheiro enviado por get
16 if(!empty($_GET['nomeFicheiroXML'])){
17     echo obterDados($_GET['nomeFicheiroXML']);
18 } else {
19     echo "Ocorreu um erro -- não foi dado um nome de ficheiro";
20 }
21 ?&gt;</pre
```



PHP – REST - JAXB

Código do pag_clienteREST.php

```
TP_JAXB_REST_sendpo > pag_clienteREST.php > ...
1 <?php
2     require("php_chamaAPI.php");
3     $url = "http://localhost/IS_WebServices/TP_JAXB_REST_sendpo/servicoREST.php";
4     $nomeFicheiro = "po.xml";
5
6     $make_call = callAPI('GET', $url . "?nomeFicheiroXML=" . urldecode($nomeFicheiro), null);
7     $response = $make_call;
8
9     header("Content-Type: text/plain");
10    echo $response;
11 ?>
```



PHP – REST - JAXB

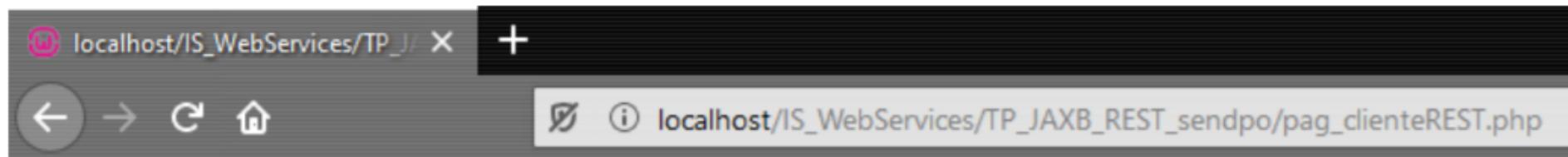
Teste POSTMAN

The screenshot shows the POSTMAN interface with the following details:

- Request Type:** GET
- URL:** http://localhost/IS_WebServices/TP_JAXB_REST_sendpo/servicoREST.php?nomeFicheiroXML=po.xml
- Params:** A table showing a single parameter: nomeFicheiroXML with value po.xml.
- Body:** A table showing a single parameter: Key with value Value.
- Status:** 200 OK
- Response:** A list of items to ship:
 - Ship the following items to:
 - Alice Smith
 - 123 Maple Street
 - Cambridge, MA 12345
 - US
 -
 - 5 copies of "Nosferatu - Special Edition (1929)"
 - 3 copies of "The Mummy (1959)"
 - 3 copies of "Godzilla and Mothra: Battle for Earth/Godzilla vs. King Ghidorah"

PHP – REST - JAXB

Teste Browser (pag_clienteREST.php)



Ship the following items to:

Alice Smith
123 Maple Street
Cambridge, MA 12345
US

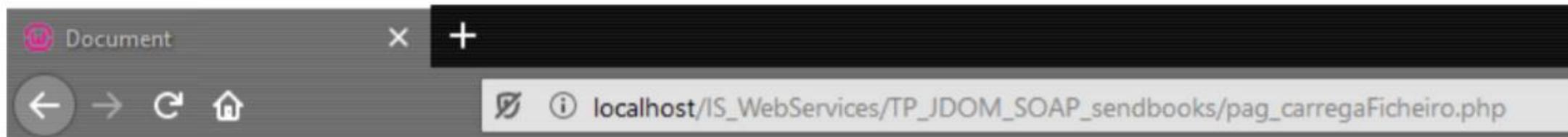
5 copies of "Nosferatu - Special Edition (1929)"

3 copies of "The Mummy (1959)"

3 copies of "Godzilla and Mothra: Battle for Earth/Godzilla vs. King Ghidora"

PHP – SOAP – JDOM com FileUpload

Página PHP para submeter o ficheiro xml:



The screenshot shows a web browser window with the following details:

- Address bar: localhost/IS_WebServices/TP_JDOM_SOAP_sendbooks/pag_carregaFicheiro.php
- Content area:
 - Text: "Selezione o ficheiro:"
 - Input field: "Explorar..." (Browse...) button followed by the message "Nenhum ficheiro selecionado." (No file selected).
 - Button: "submeter" (Submit)

PHP – SOAP – JDOM com FileUpload

Código:

```
TP_JDOM_SOAP_sendbooks > 🌐 pag_carregaFicheiro.php
● 1   <!DOCTYPE html>
2   <html lang="en">
3     <head>
4       <?php
5         header("Access-Control-Allow-Origin: *");
6         header("Access-Control-Allow-Methods: OPTIONS, GET, POST, PUT, DELETE");
7         header("Access-Control-Allow-Headers: Content-Type");
8       ?>
9       <meta charset="UTF-8">
10      <meta name="viewport" content="width=device-width, initial-scale=1.0">
11      <meta http-equiv="X-UA-Compatible" content="ie=edge">
12      <title>Document</title>
13      <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
14    </head>
15    <body>
16
17      <form action="upload.php" enctype="multipart/form-data" method="POST">
18        Selecione o ficheiro:
19        <br>
20        <input type="file" name="ficheiro" />
21        <br><br>
22        <input type="submit" value="submeter" />
23      </form>
24    </body>
25  </html>
```



PHP – SOAP – JDOM com FileUpload

Código para upload:

```
TP_JDOM_SOAP_sendbooks > upload.php > ...
1 ?php
2     header("Content-type: text/xml");
3     include("clienteSOAP.php");
4
5     if(!isset($_FILES['ficheiro'])){
6         exit('Ficheiro não enviado');
7     }
8     // verificar erro
9     if($_FILES['ficheiro']['error']){
10        exit('Erro no upload');
11    }
12    // -- tamanho do ficheiro
13    if($_FILES['ficheiro']['size'] > 40000000){
14        exit("ficheiro demasiado grande");
15    }
16    // -- tipo ficheiro
17    if($_FILES['ficheiro']['type'] != "text/xml"){
18        exit("tipo de ficheiro não suportado");
19    }
20    $name = $_FILES['ficheiro']['name'];
21    $ficheiro_final = $name;
22
23    if(file_exists($ficheiro_final)){
24        //-- tocar ficheiro
25        unlink($ficheiro_final);
26    }
27
28    if(!move_uploaded_file($_FILES['ficheiro']['tmp_name'], $ficheiro_final)){
29        exit("Ocorreu um erro ao carregar!");
30    }
31    echo '<?xml version="1.0" encoding="ISO-8859-1"?>';
32    echo clienteSOAP($ficheiro_final);
```

PHP – SOAP – JDOM com FileUpload

Código da function clienteSOAP(string nomeFicheiro):

```
<?php
require_once "nusoap.php";
function clienteSOAP($nomeFicheiroCarregado = "books.xml"){

    $client = new nusoap_client("http://localhost/IS_WebServices/TP_JDOM_SOAP_sendbooks/servicoSOAP.php");
    $error = $client->getError();
    if ($error) {
        echo "<h2>Constructor error</h2><pre>" . $error . "</pre>";
    }

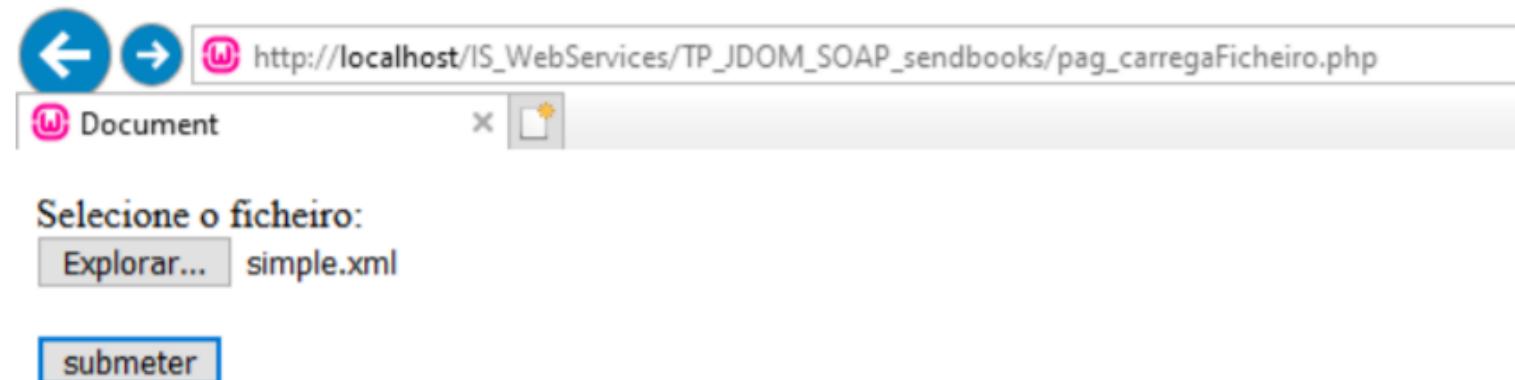
    $result = $client->call("obterDados", array("nome" => $nomeFicheiroCarregado));

    if ($client->fault) {
        return "<h2>Fault</h2><pre>" . $result . "</pre>";
    }
    else {
        $error = $client->getError();
        if ($error) {
            return "<h2>Error</h2><pre>" . $error . "</pre>";
        }
        else {
            return $result;
        }
    }
}

?>
```

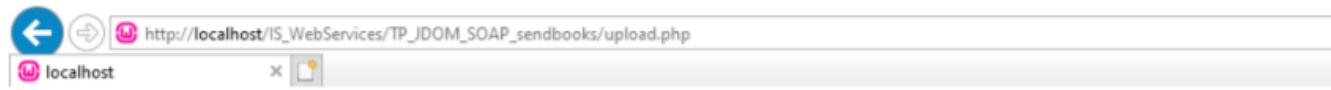
PHP – SOAP – JDOM com FileUpload

Teste Upload e visualização com o simple.xml:



PHP – SOAP – JDOM com FileUpload

Teste Upload e visualização com o simple.xml:



```
<?xml version="1.0" encoding="ISO-8859-1"?>
- <breakfast_menu>
  - <food>
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
    <calories>650</calories>
  </food>
  - <food>
    <name>Strawberry Belgian Waffles</name>
    <price>$7.95</price>
    <description>Light Belgian waffles covered with strawberries and whipped cream</description>
    <calories>900</calories>
  </food>
  - <food>
    <name>Berry-Berry Belgian Waffles</name>
    <price>$8.95</price>
    <description>Light Belgian waffles covered with an assortment of fresh berries and whipped cream</description>
    <calories>900</calories>
  </food>
  - <food>
    <name>French Toast</name>
    <price>$4.50</price>
    <description>Thick slices made from our homemade sourdough bread</description>
    <calories>600</calories>
  </food>
  - <food>
    <name>Homestyle Breakfast</name>
    <price>$6.95</price>
    <description>Two eggs, bacon or sausage, toast, and our ever-popular hash browns</description>
    <calories>950</calories>
  </food>
```