FICHA PRÁTICA n.º 9

Integração de Sistemas de Informação

Unidade Curricular:	Integração de Sistemas de Informação

Tema da Ficha Pratica:

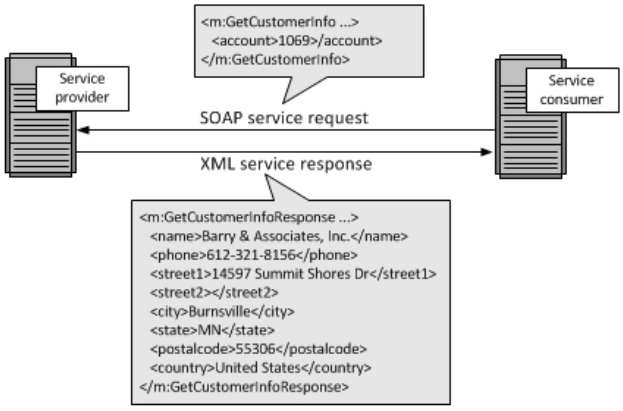
Este tutorial/ficha pretende apresentar a interligação entre as seguintes componentes: XML e webservices (implementados via SOAP e REST) e com implementação em Java e PHP.

Objectivos:

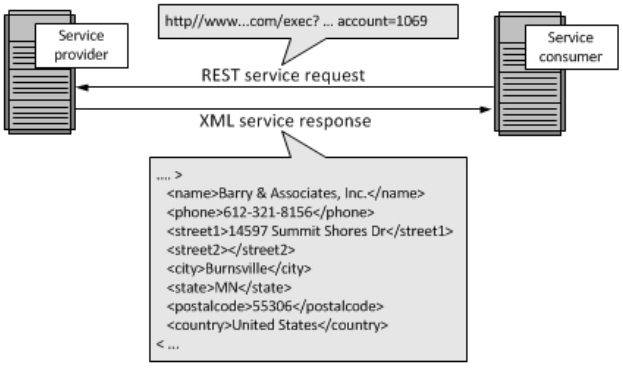
Pretende-se com esta ficha prática que os alunos:

1) implementem ou usem a implementação feita na ficha prática anterior um parsing distribuído cliente/servidor de ficheiros XML (de tamanho reduzido e de grande volume);
2) Criem uma tabela numa base de dados (ex. em mySQL), ler um ficheiro XML e importar os dados para essa tabela e executar queries através de web services SOAP e REST (ver figura seguinte). A implementação neste tutorial será efetuada usando as linguagens de programação Java e PHP.

SOAP.



Representation State Transfer (REST)



Fonte: https://www.service-architecture.com/articles/web-services/web_services_explained.html

Conteúdo

Bibliografia

Para apoio a esta ficha os alunos devem consultar os apontamentos teóricos e práticos da disciplina bem como de outros recursos online, como por exemplo:

a) Validação de Ficheiros XML:

http://www.w3schools.com/xml/xml_validator.asp

http://www.xmlvalidation.com/

http://validator.w3.org/

b) Informação sobre a interpretação de ficheiros XML usando programação:

http://www.developerfusion.com/code/2064/a-simple-way-to-read-an-xml-file-in-java/

http://docs.oracle.com/javase/tutorial/jaxp/stax/example.html

http://docs.oracle.com/cd/B28359_01/appdev.111/b28394/adx_j_parser.htm

http://www.mkyong.com/tutorials/java-xml-tutorials/

https://www.tutorialspoint.com/java_xml/

c) Parsers XML:

- Usando a linguagem de programação JAVA:

  1.Document Object Model (DOM):

  http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html

  2. Simple API for XML (SAX):

  http://docs.oracle.com/javase/1.5.0/docs/api/org/xml/sax/package-summary.html

  http://www.saxproject.org/quickstart.html

  3. JDOM XML Parser

  http://www.jdom.org/

  http://www.jdom.org/docs/oracle/jdom-part1.pdf

  4. Java API for XML Processing (JAXP):

  http://www.oracle.com/technetwork/java/intro-140052.html

  5. Java Architecture for XML Binding (JAXB)

  http://jaxb.java.net/tutorial/index.html

- Usando a linguagem de Programação PHP:

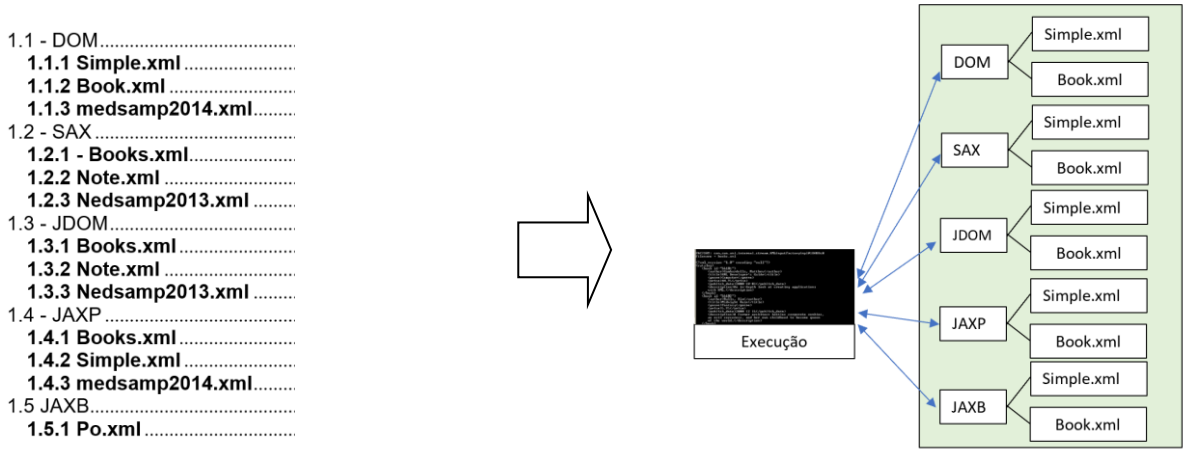
  https://www.php.net/xml

  https://www.php.net/manual/en/refs.webservice.php

Nota:

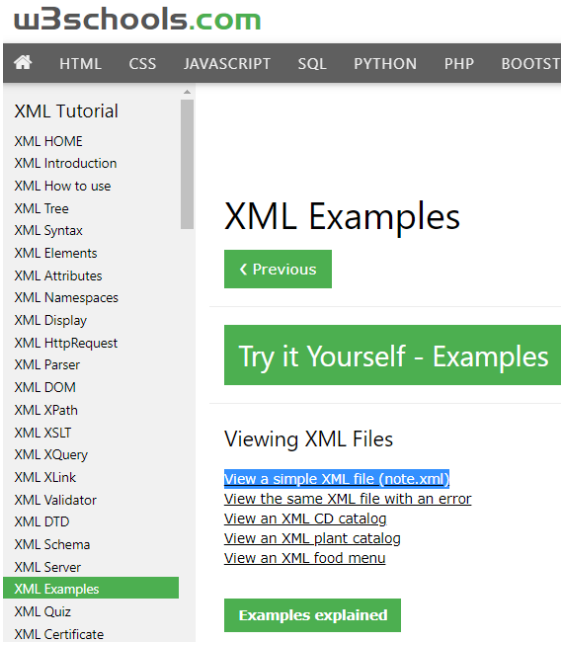
- Servirão como guia e apoios deste trabalho, a ficha prática (Java com XML) e links disponibilizados no moodle da unidade curricular.
- Para simplificar o processo, pretende-se que o servidor receba o ficheiro XML, faça o parsing do ficheiro na estrutura de armazenamento implementada (DOM, SAX, JDOM, JAXP e JAXB) e que faça o print do ficheiro do lado do servidor.

Em termos gerais, pretendeu-se através da ficha prática anterior, executar os parsers XML, usando a linguagem de programação JAVA:

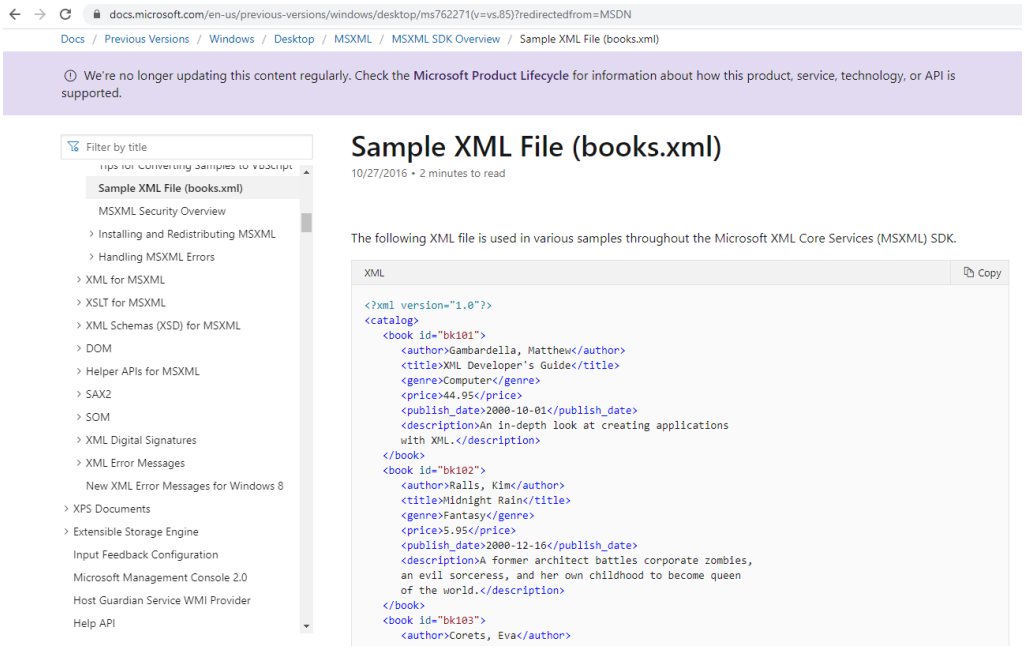


Os ficheiros XML a submeter ao servidor via cliente, são os seguintes:

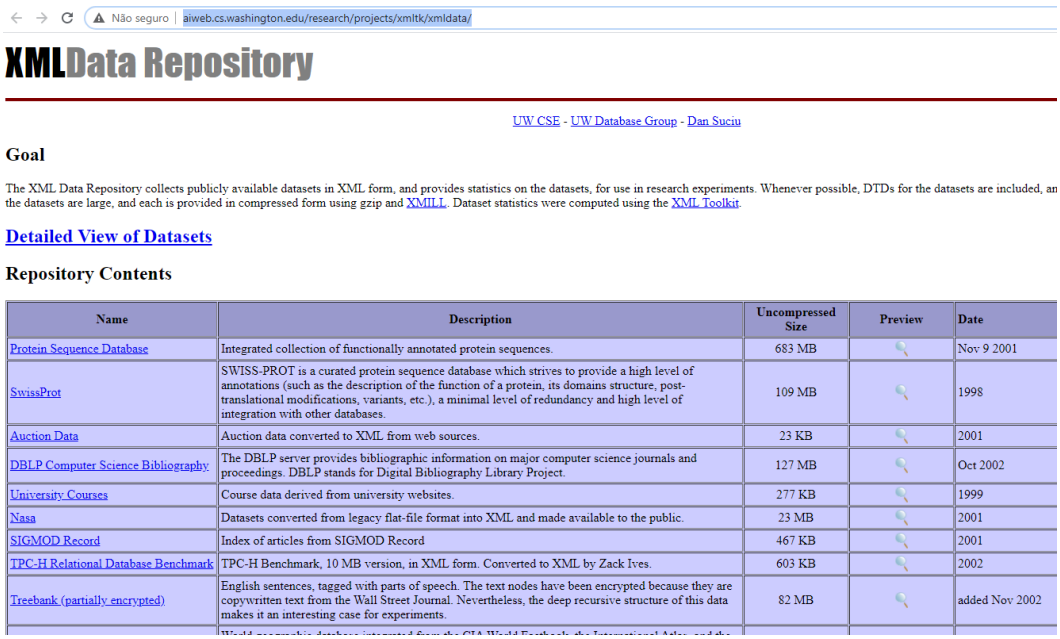
1. Um ficheiro simples obtido em: http://www.w3schools.com/xml/xml_examples.asp



2. O ficheiro Books.xml: http://msdn.microsoft.com/en-us/library/ms762271(v=vs.85).aspx



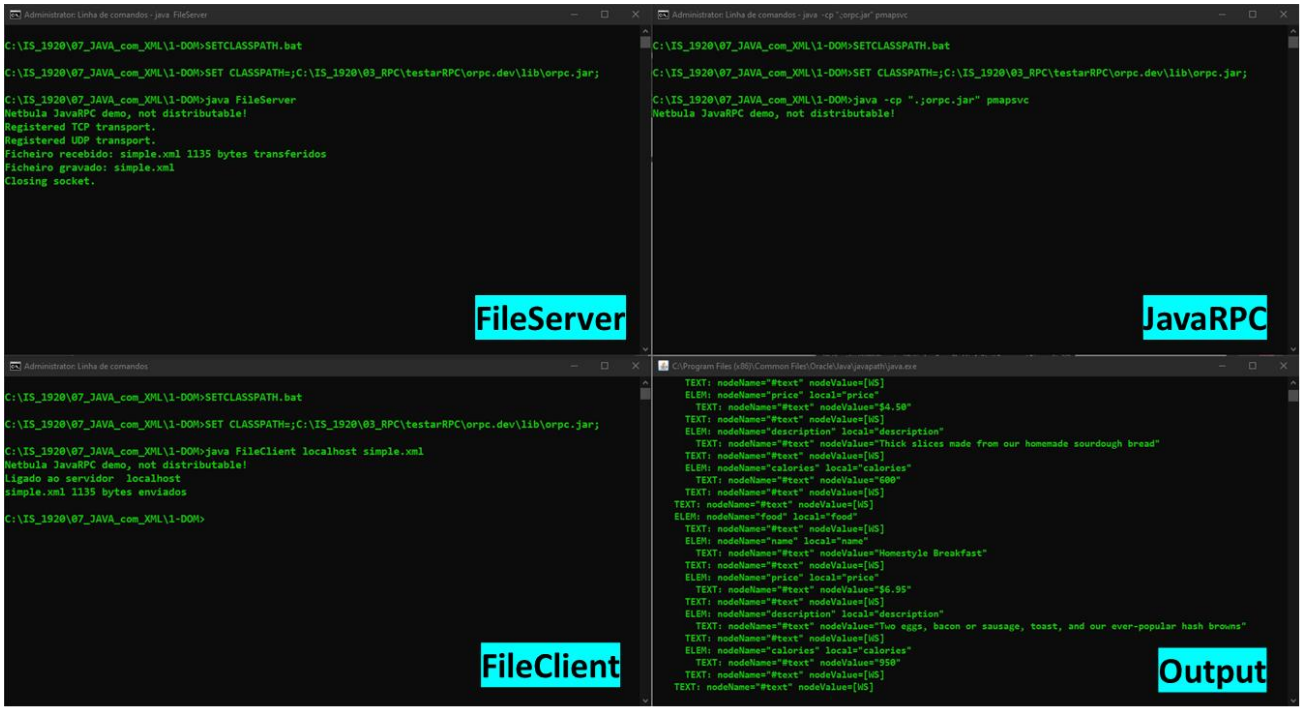
3. Um ficheiro de grande volume obtido em: http://aiweb.cs.washington.edu/research/projects/xmltk/xmldata/



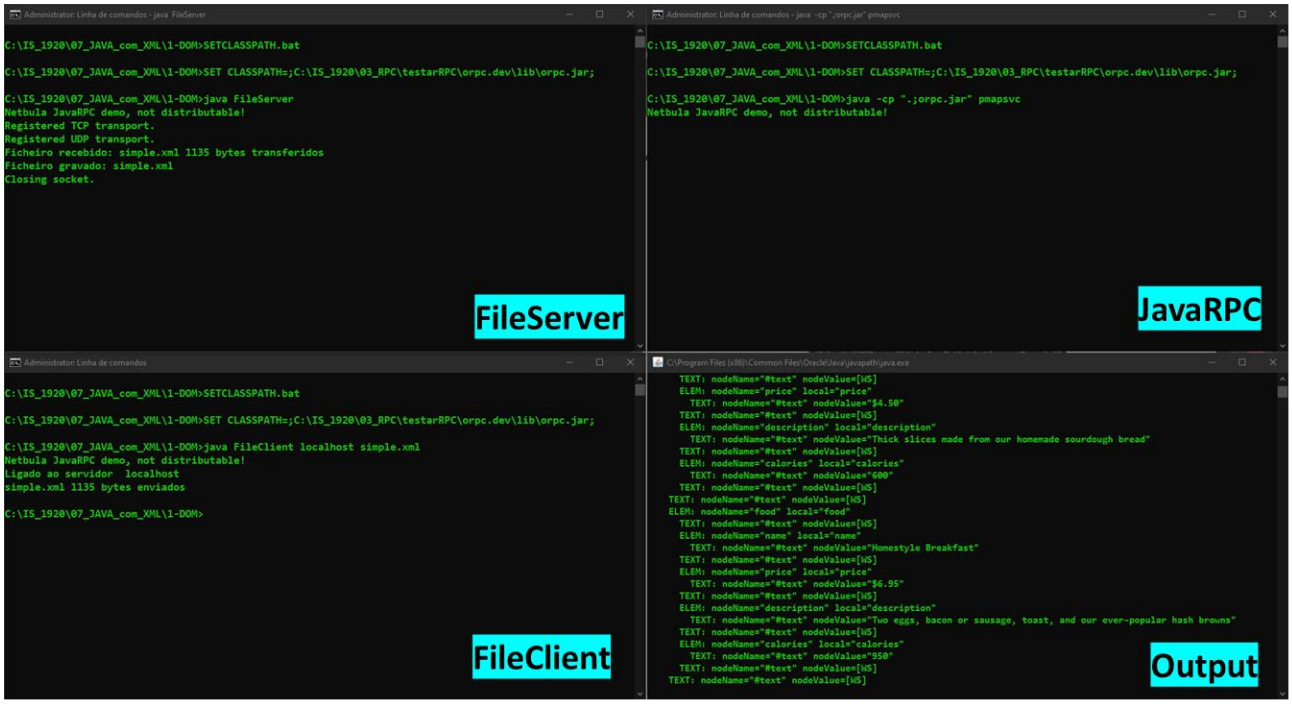
1. Resumo da implementação do tutorial do parser de XML usando Java

1.1 – Parser DOM

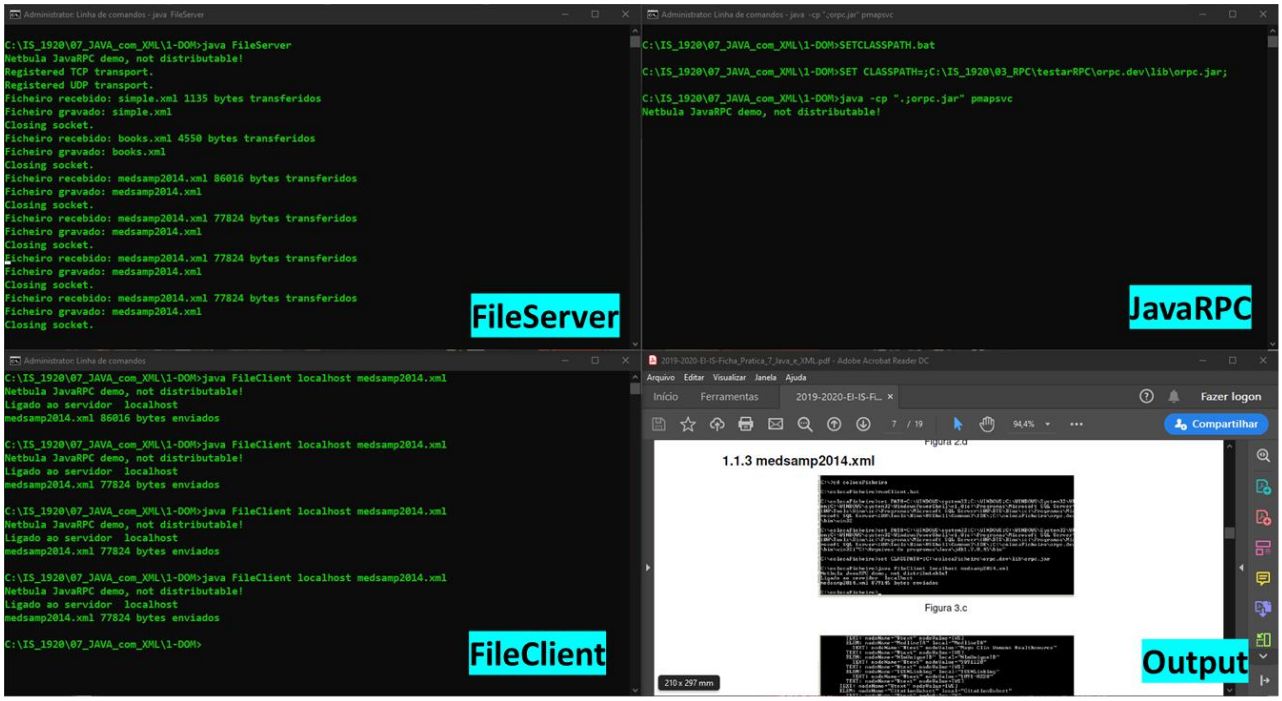
1.1.1 Com o ficheiro Simple.xml



1.1.2 Com o ficheiro books.xml

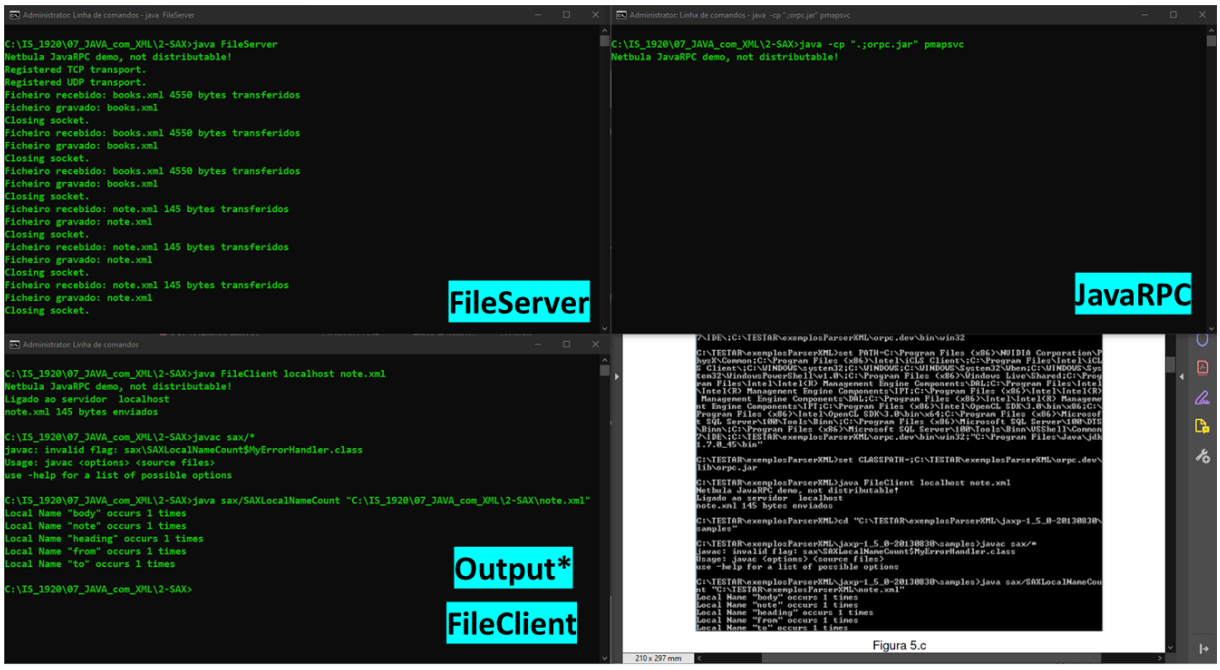


1.1.2 Com o ficheiro medsamp2014.xml



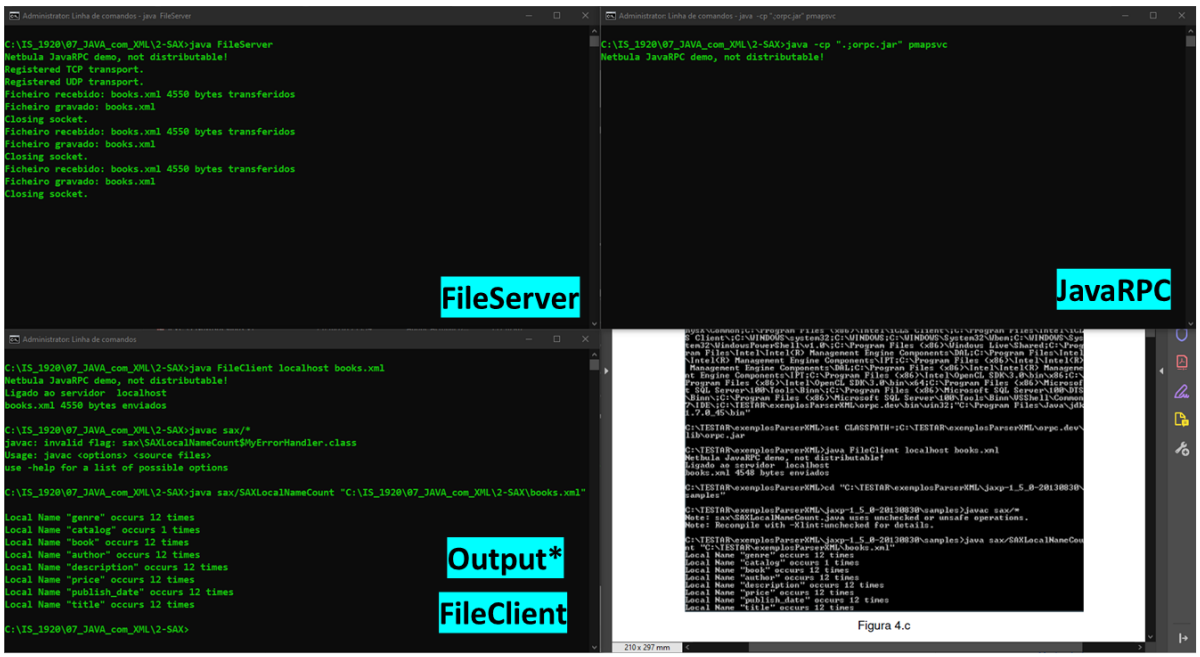
1.2 – Parser SAX

1.2.1 Com o ficheiro note.xml



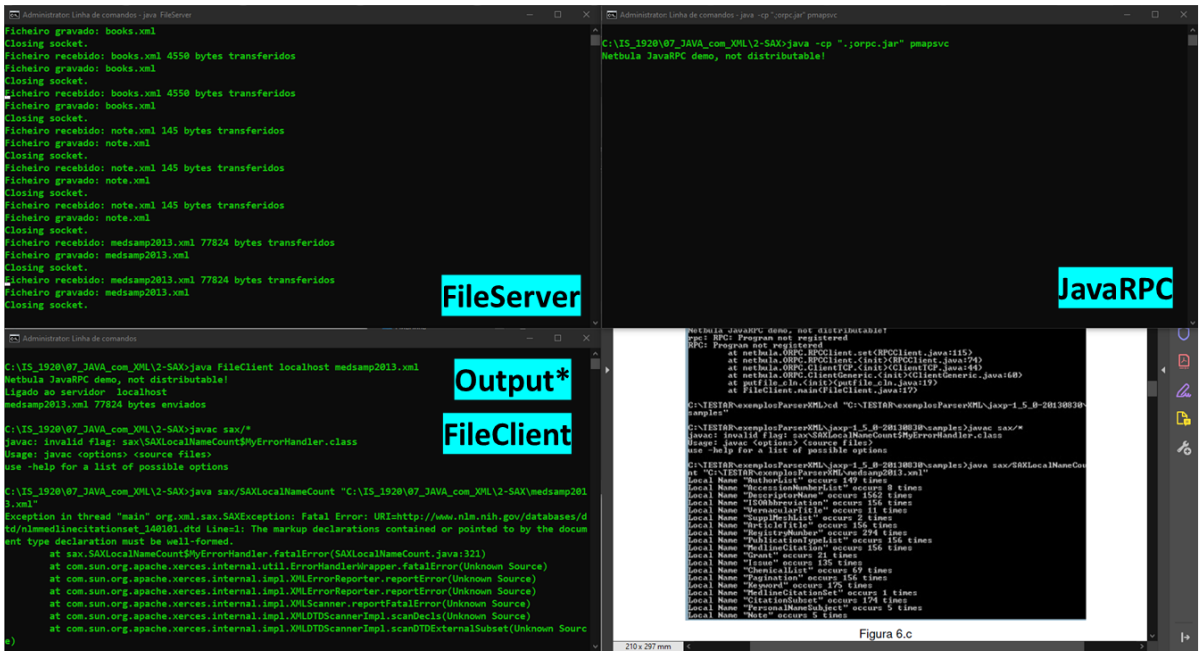
***SAXLocalNameCount.java conta quantas vezes estava presente cada elemento no ficheiro**

1.2.2 Com o ficheiro books.xml



***SAXLocalNameCount.java conta quantas vezes estava presente cada elemento no ficheiro**

1.2.3 Com o ficheiro medsamp2013.xml



***ERRO DE EXECUÇÃO: DTD do ficheiro com notações "incorretas". Desafio: Necessário analisar a sintaxe do DTD.**

1.3 – Parser JDOM

1.3.1 Com o ficheiro note.xml

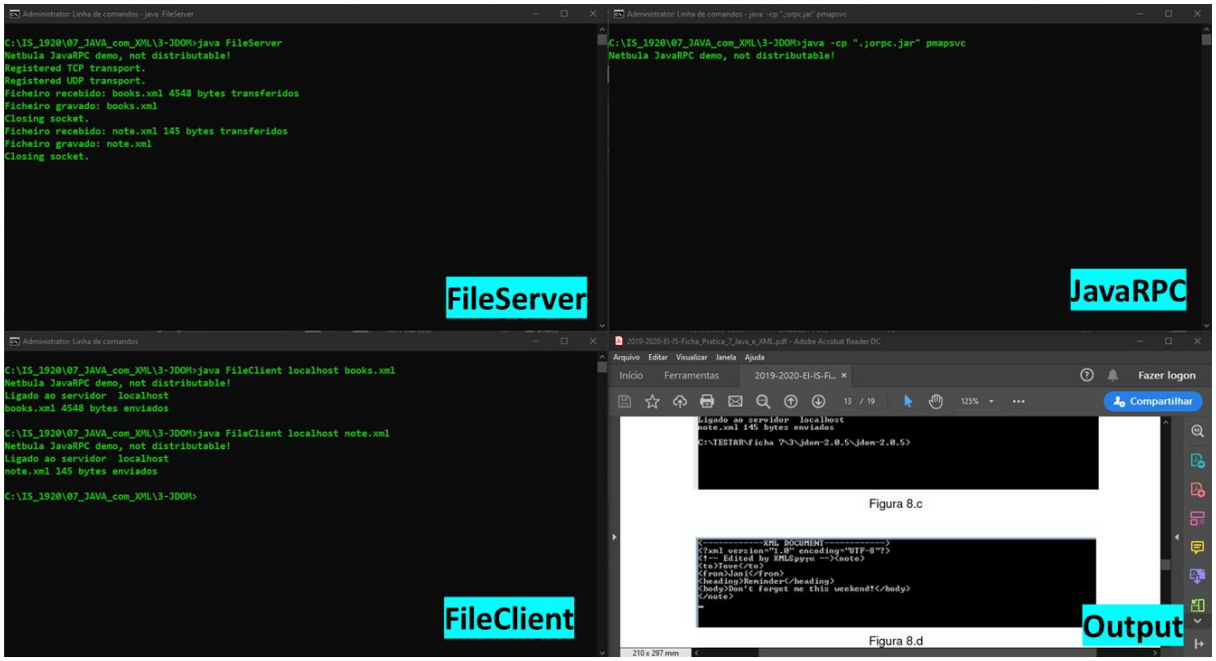


Figura 8.c

Figura 8.d

1.3.2 Com o ficheiro books.xml

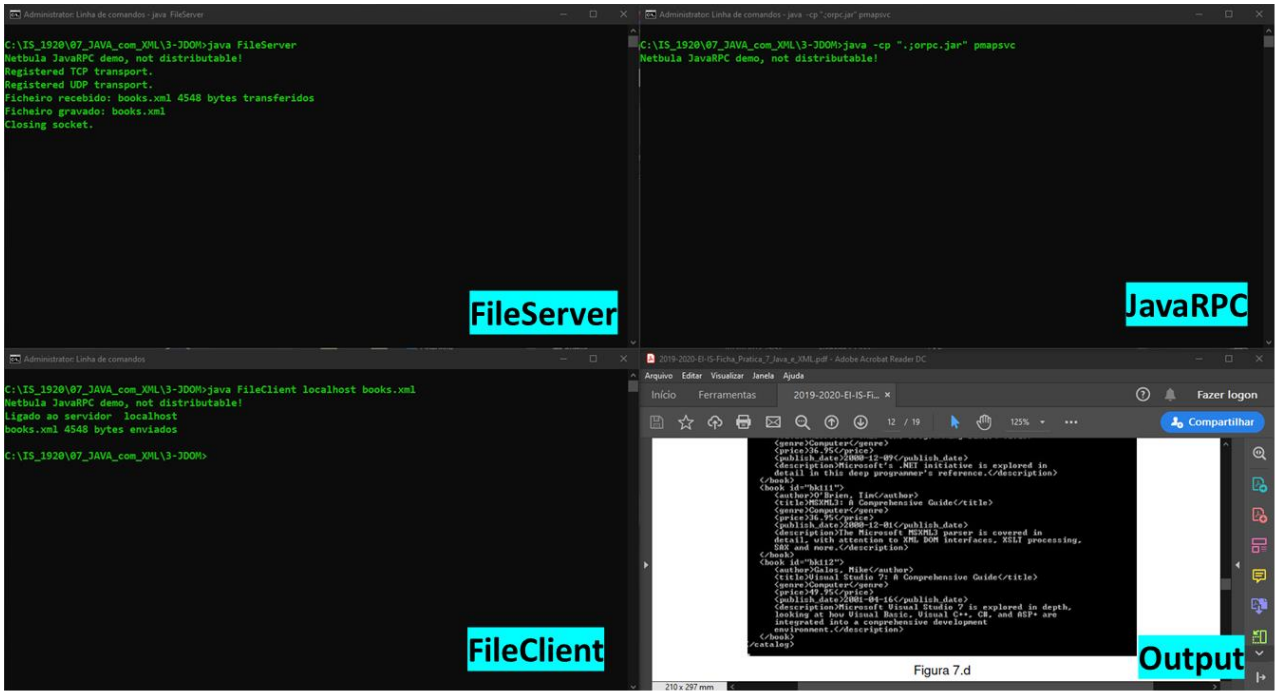


Figura 7.d

1.3.3 Com o ficheiro medsamp2013.xml

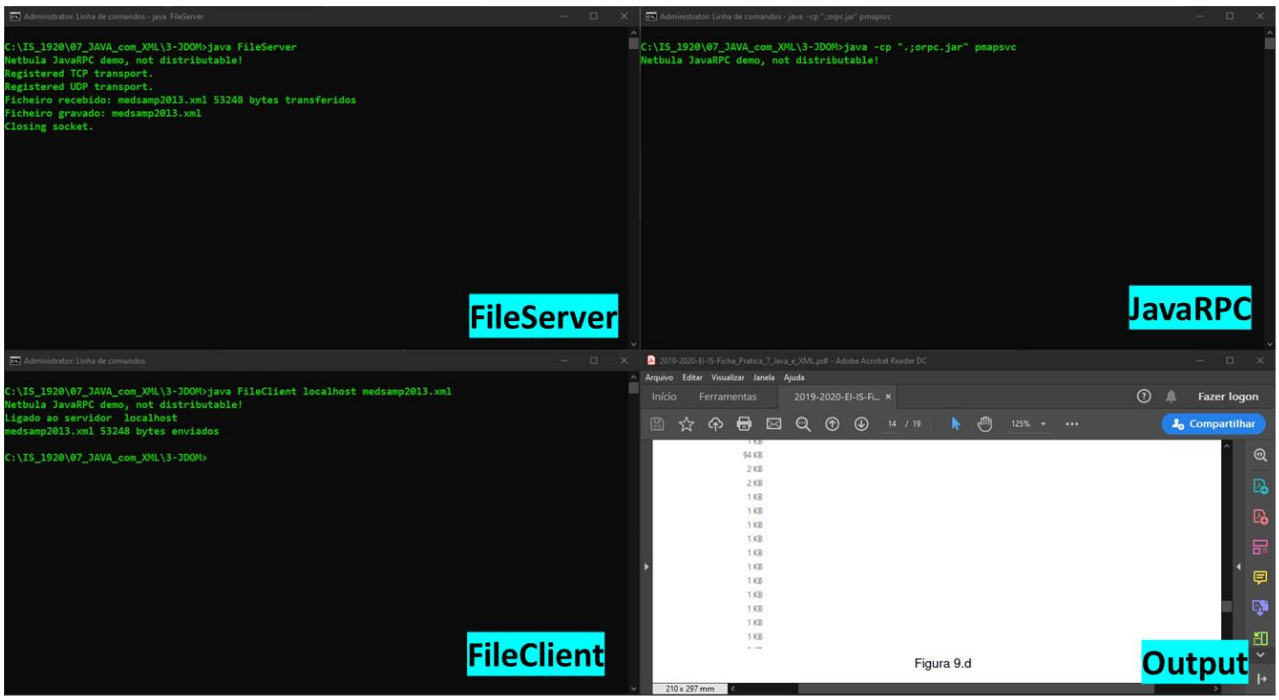
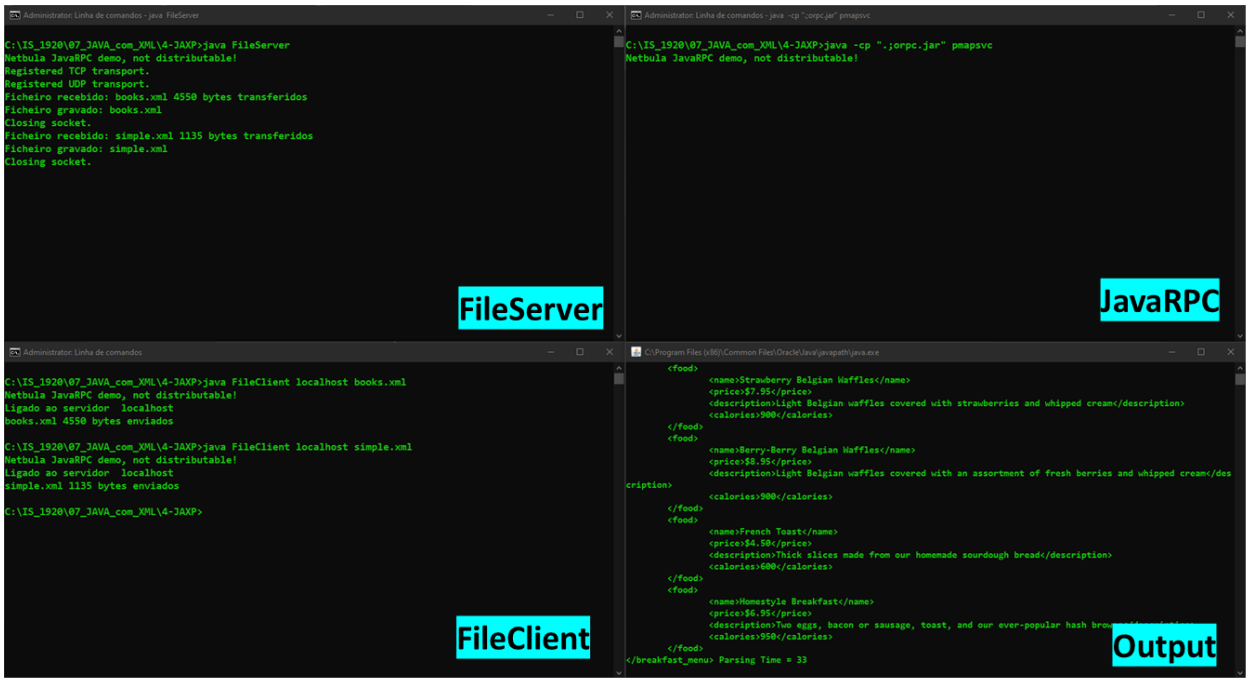


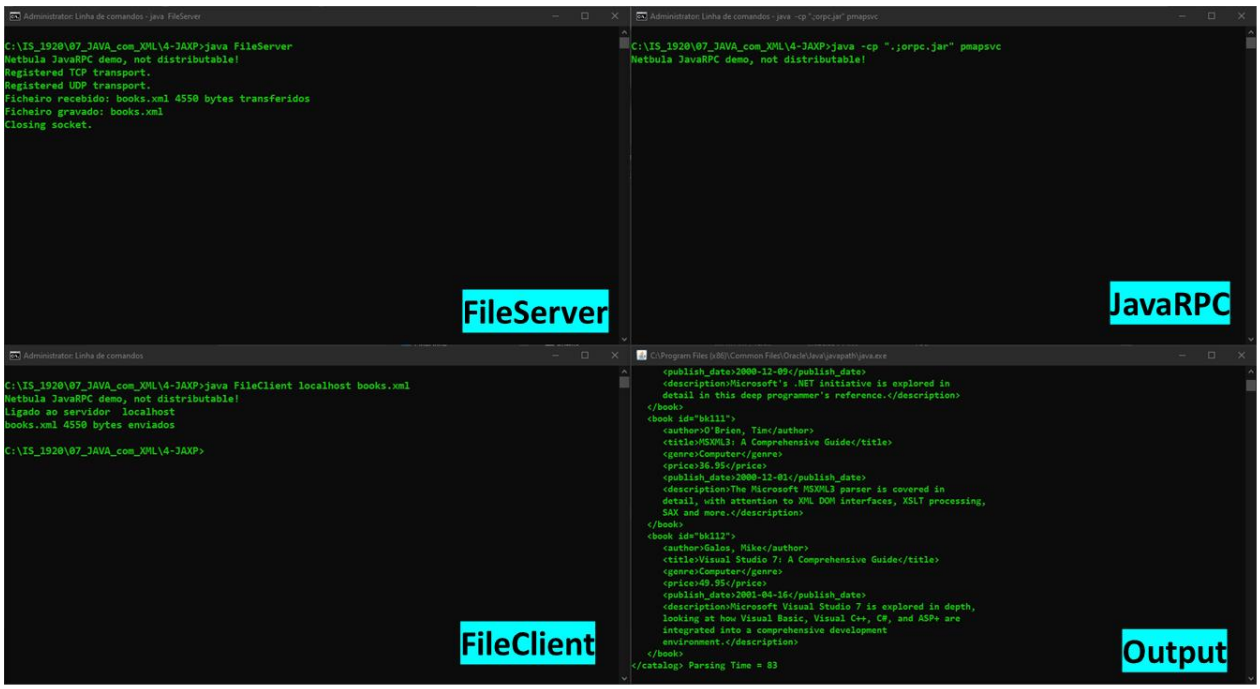
Figura 9.d

1.4 – Parser JAXP

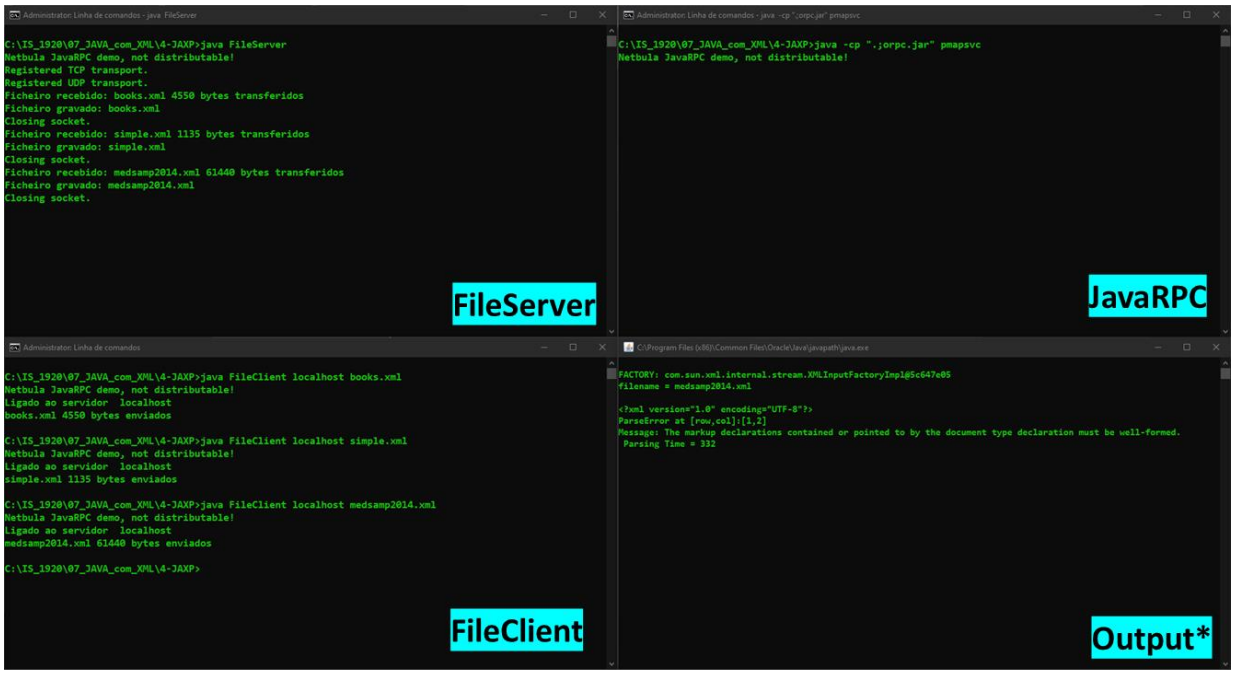
1.4.1 Com o ficheiro simple.xml



1.4.2 Com o ficheiro books.xml

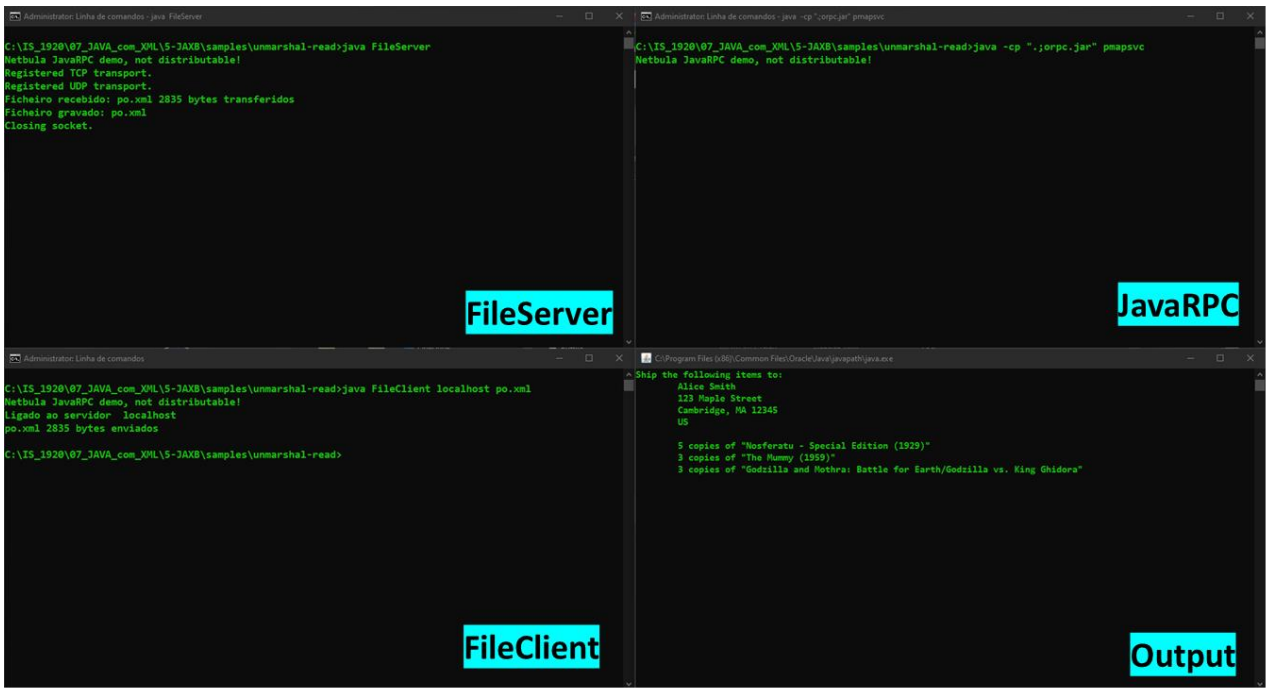


1.4.3 Com o ficheiro medsamp2014.xml



***ERRO DE EXECUÇÃO: DTD do ficheiro com notações "incorretas". Desafio: Necessário analisar a sintaxe do DTD.**

1.4.4 Com o ficheiro po.xml



2. Implementação de Webservices SOAP e REST integrando ficheiros XML em Base de Dados usando a abordagem Cliente/Servidor

Sugere-se a utilização das seguintes ferramentas/ambiente de desenvolvimento:

- **Netbeans IDE** 8.2 ALL (JAVA WebServices); **Glassfish** (Java WebServices – Servidor WEB); **Visual Studio Code** (PHP WebServices); **Plataforma WAMP** – 3.1.9 64bit (PHP WebServices) + Apache 2.4.39 + Mysql 5.7.26 ◦ PHP 7.2.18; **Testes:** ◦ **SOAP UI** ◦ **POSTMAN** ◦ **BROWSER** (F12 Modo Developer)

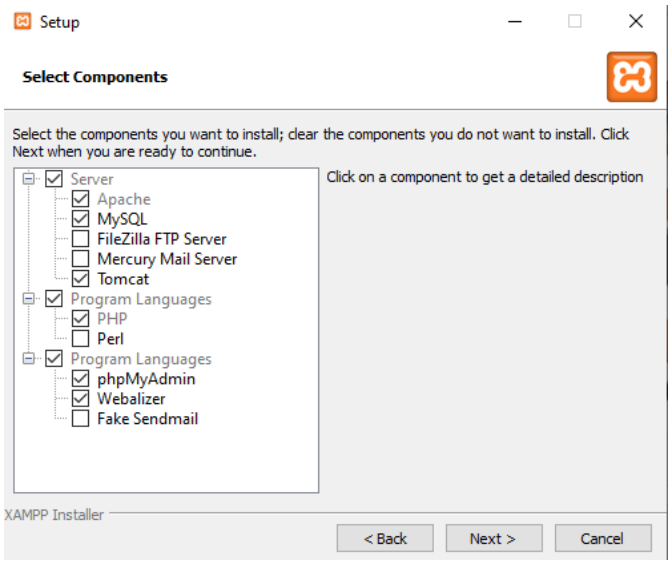
2.1 Enquadramento

Com base nos conhecimentos adquiridos da implementação de um parser em JAVA (com várias abordagens), pretende-se que se criem WebServices (SOAP e REST) implementados em Java e em PHP para receber no lado do servidor um ficheiro XML e o importar para uma tabela de uma base de dados.

Neste contexto, para:

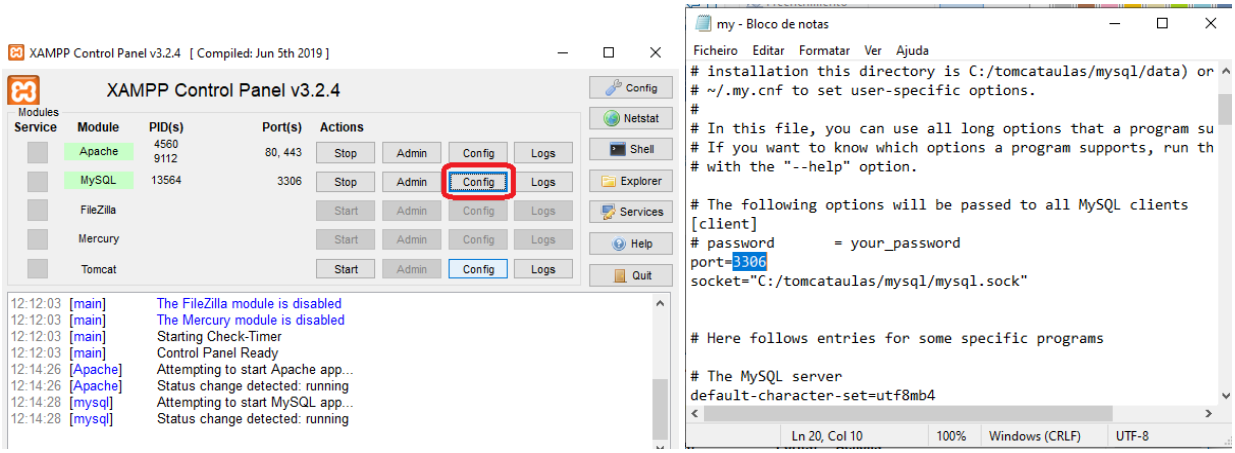
- a implementação dos Webservices em Java iremos utilizar o IDE – Interface Desktop Environment netBeans com o módulo “mysql-connector-java-8.0.18.jar” para fazer a ligação à base de dados. Este ficheiro jar deverá estar na diretoria “build\web\WEB-INF\lib”.

  NOTA: As configurações de acesso à base de dados estão definidas no ficheiro “IS_TP1_JAVA\src\java\com\is\db\Alunos_ws_rest.java”.
- a implementação dos webservices em PHP iremos ter de instalar um servidor WEB. Neste caso instalaremos o XAMPP, disponível em: https://www.apachefriends.org/index.html

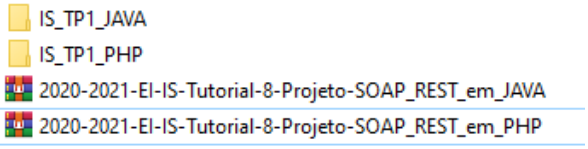


NOTA: Para simplificar a execução deste tutorial, assume-se que o servidor WEB irá ser executado **em localhost:3306** e que são definidas na base de dados mySQL as credenciais **username**:userdb **password**:teste (*em vez de username:root e password:root*).

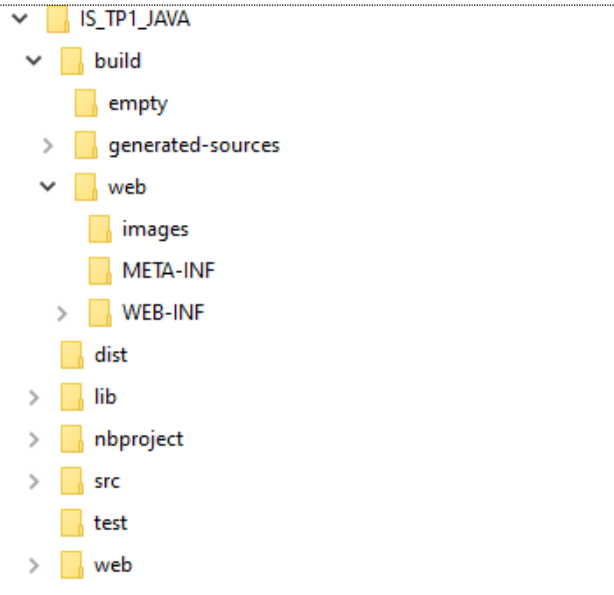
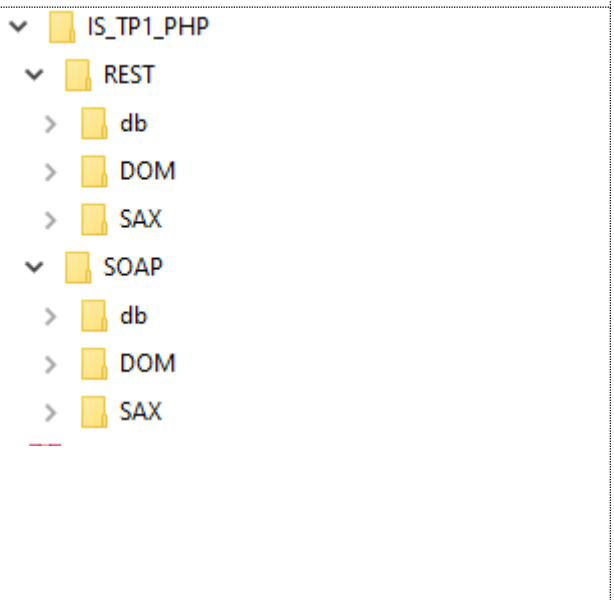
NOTA: A porta é a 3306 uma vez que por defeito o mySQL fica à escuta nessa porta. Poderá ver as configurações (ou alterar) através da configuração do ficheiro “my.ini” através do botão “Config”:



Considere os ficheiros ZIP com o código de implementação, “2020-2021-EI-IS-Tutorial-8-Projeto-SOAP_REST_em_JAVA.zip”. e “2020-2021-EI-IS-Tutorial-8-Projeto-SOAP_REST_em_PHP.zip”:

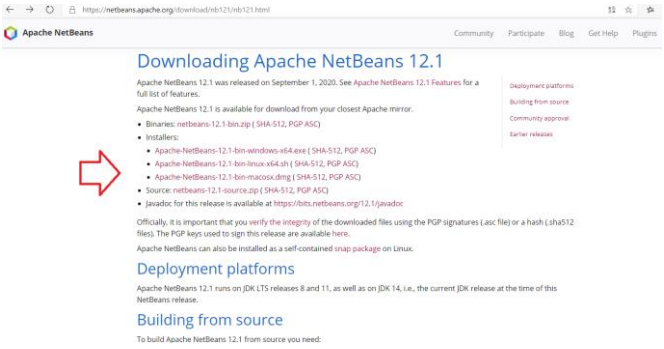


Descompacte os ficheiros dentro de uma diretoria (ex. tutorial_8), as quais deverão ter a seguinte estrutura:

Em JAVA:	Em PHP:
	

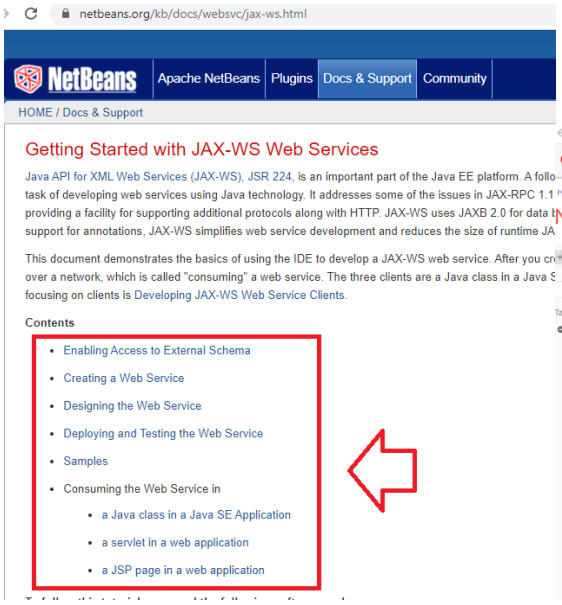
2.2 Implementação em JAVA

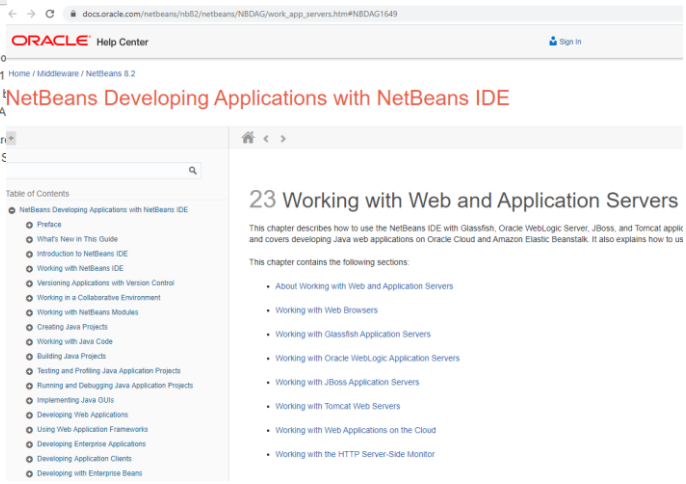
Para facilitar a edição dos ficheiros e a sua execução utilizou-se o IDE – Interface Desktop Environment NetBeans, disponível em: https://netbeans.org/ .

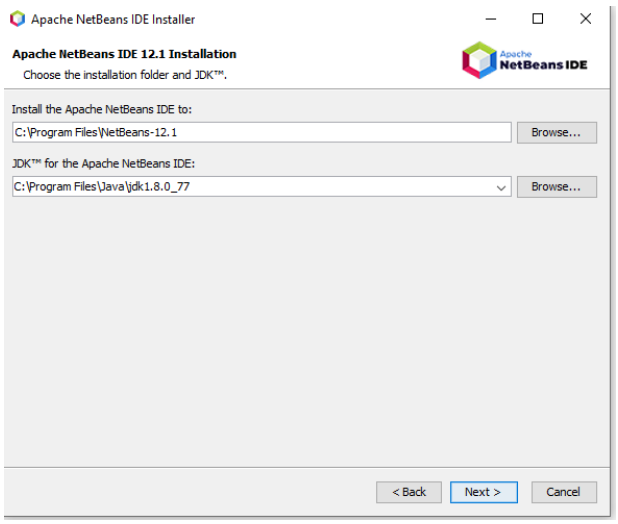


Recomenda-se a leitura desta informação:

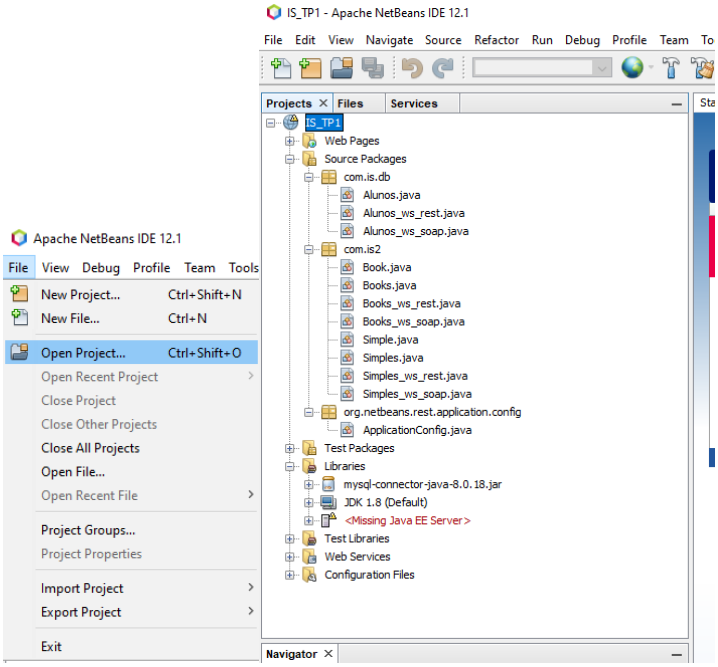
- Getting Started with JAX-WS Web Services: https://netbeans.org/kb/docs/websvc/jax-ws.html
- NetBeans Developing Applications with NetBeans IDE: https://docs.oracle.com/netbeans/nb82/netbeans/NBDAG/work_app_servers.htm#NBDAG2198
- Packaging and Deploying Desktop Java Applications: https://netbeans.org/kb/articles/javase-deploy.html#Exercise_1



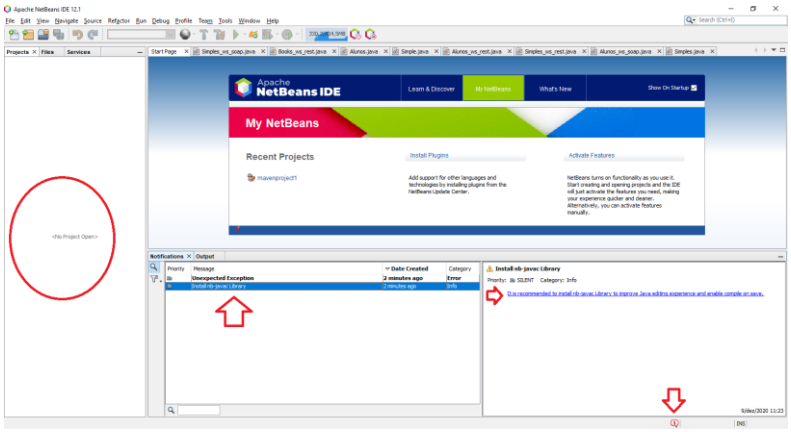




Importação do Projeto:

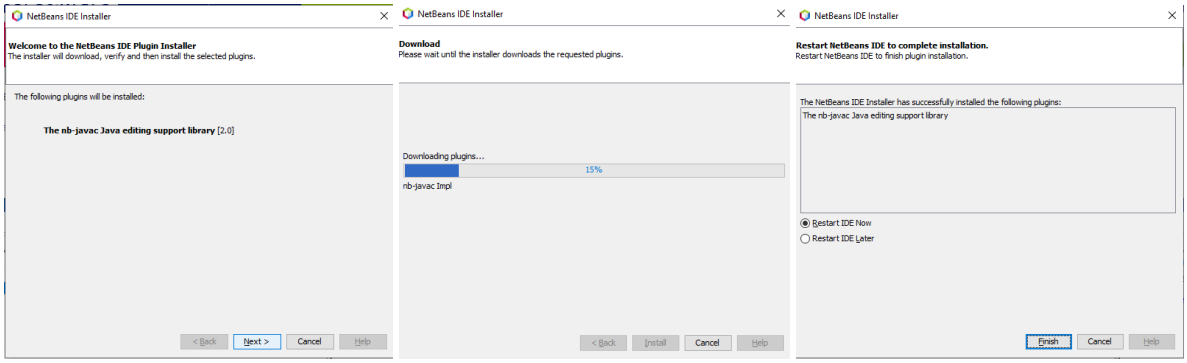


Se o projeto não foi importado, ver os erros de instalação dos plugins:

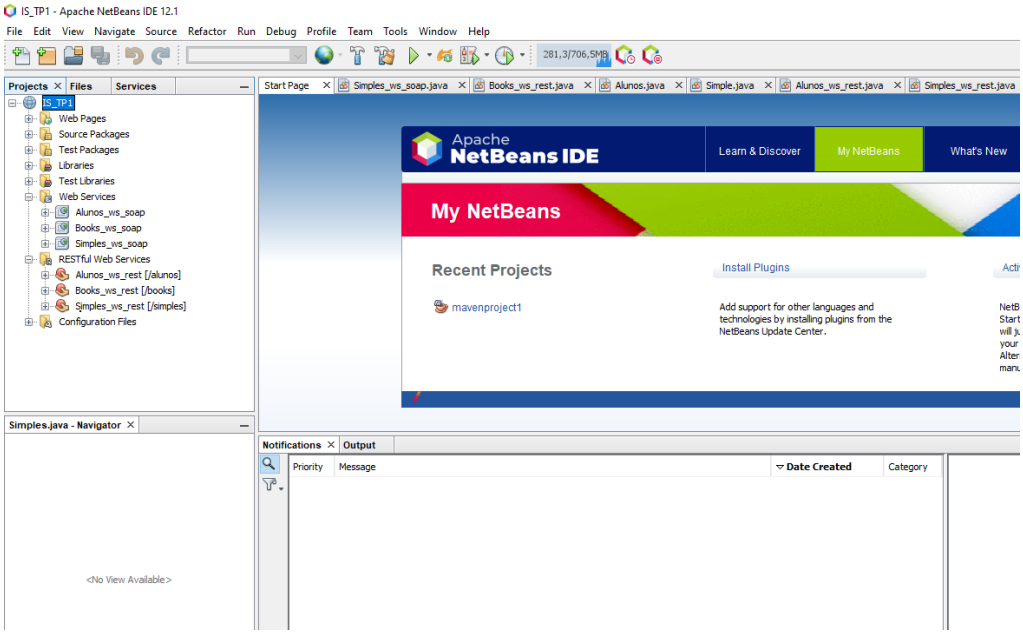


Exemplo de necessidade de instalação: Is needed to install the “nb-javac Java editing support library “

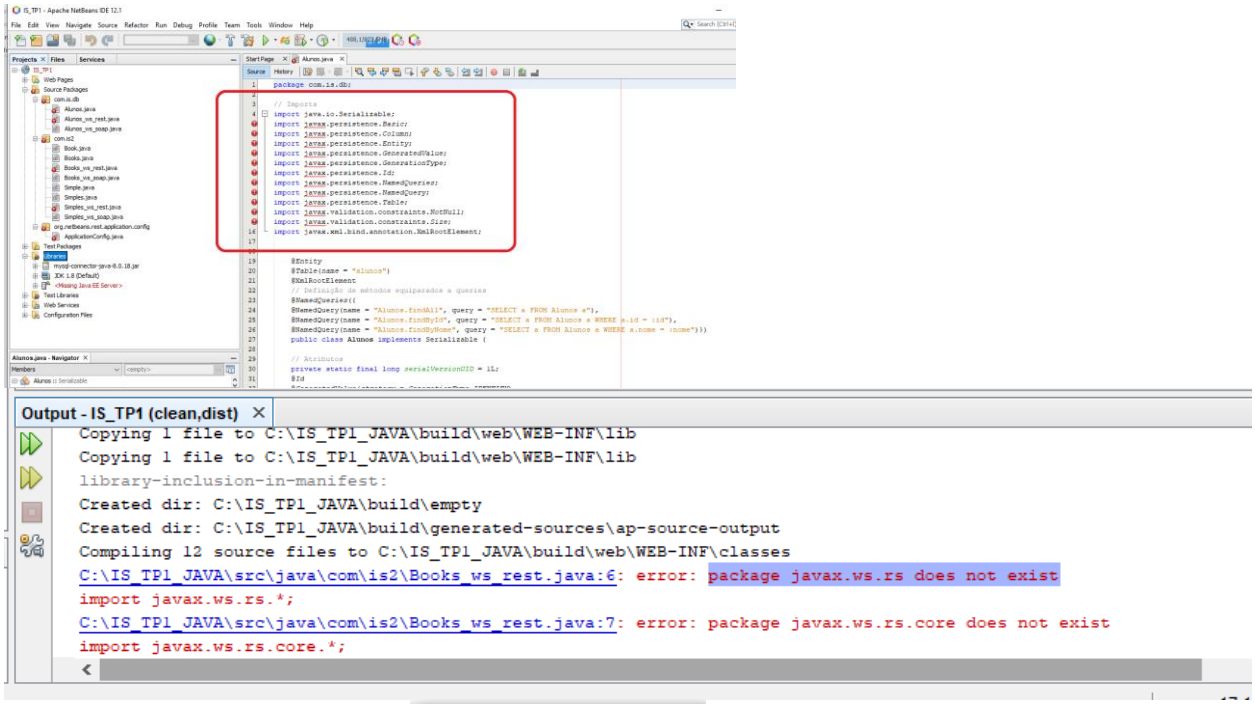
Selecione a linha “Install nb-javac library” and in the link “It is recommended to install…”

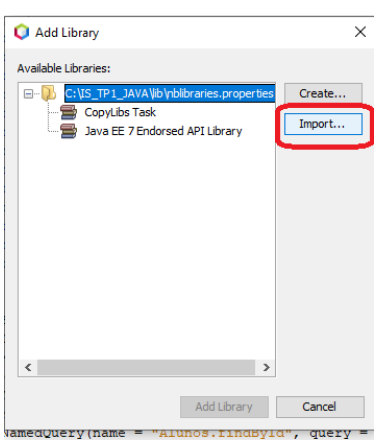


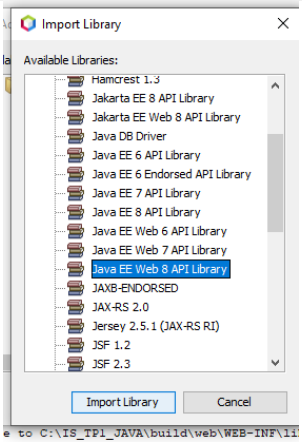
Depois, import novamente o projeto:



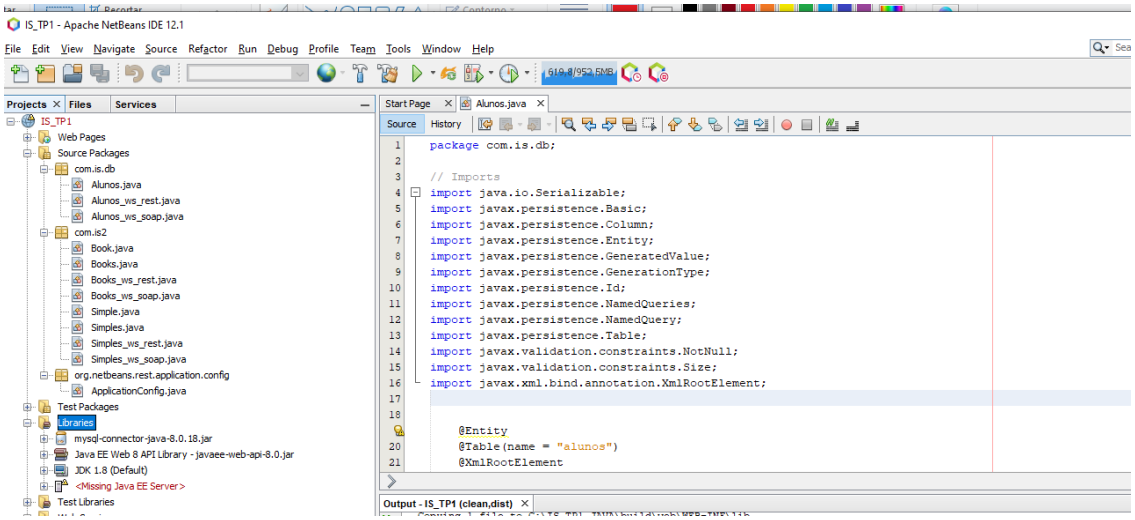
Atualização das Librarias/Packages:



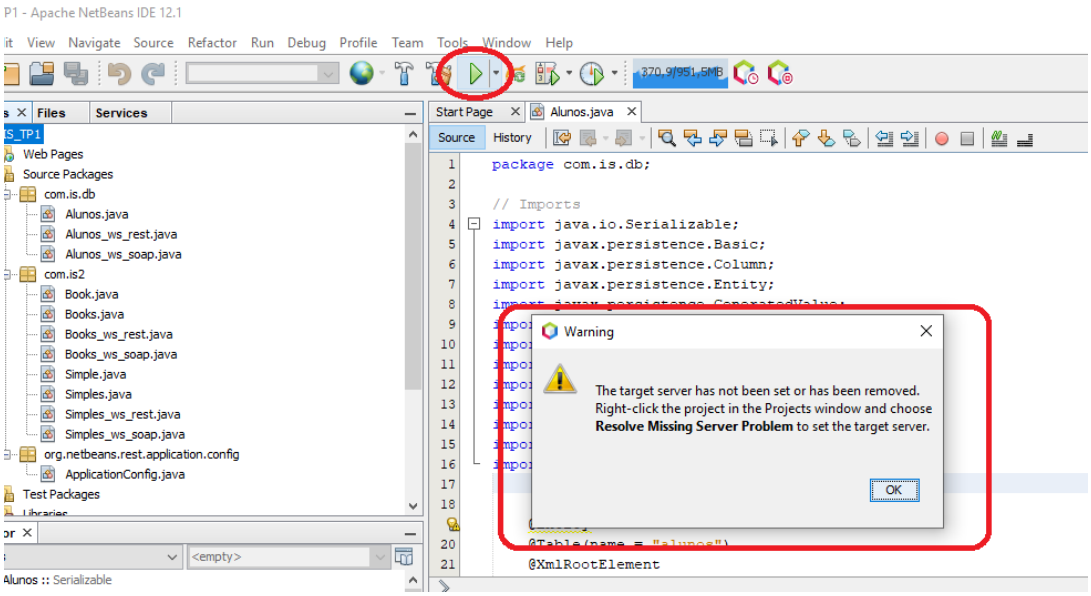


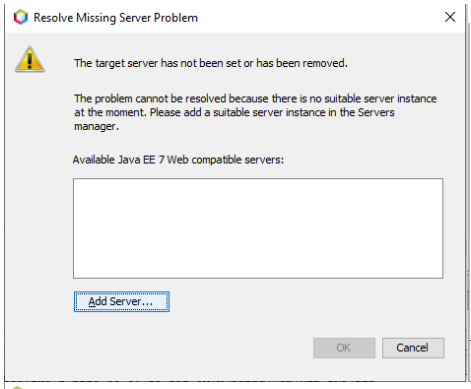


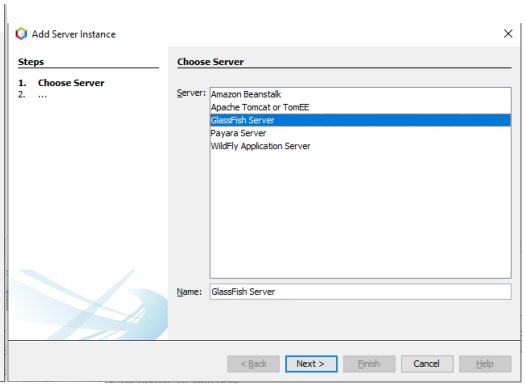
Importando o package Java EE API Library:

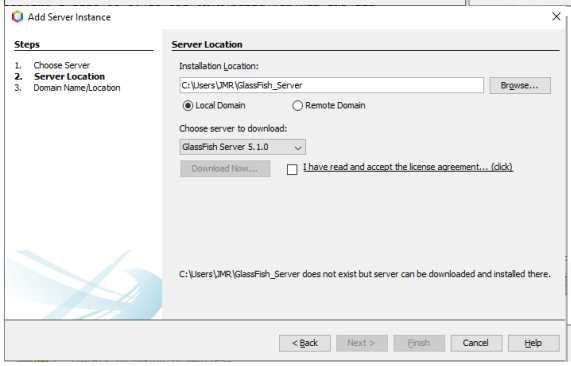


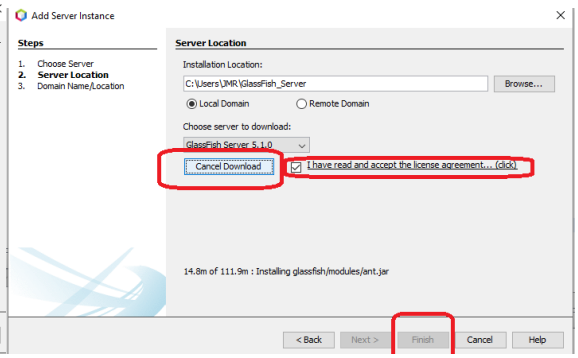
Preparando o arranque do projeto:

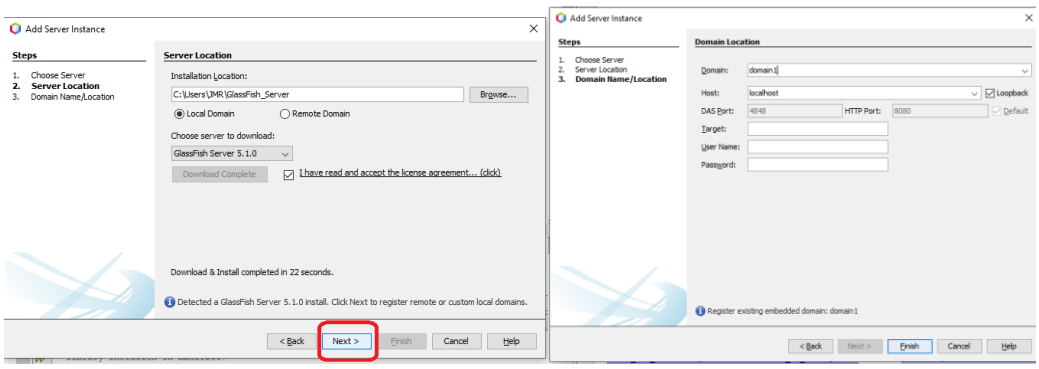

P1 - Apache NetBeans IDE 12.1
Files
Services
Web Pages
Source Packages
com.is.db
Alunos.java
Alunos_ws_rest.java
Alunos_ws_soap.java
com.is2
Book.java
Books.java
Books_ws_rest.java
Books_ws_soap.java
Simple.java
Simples.java
Simples_ws_rest.java
Simples_ws_soap.java
org.netbeans.rest.application.config
ApplicationConfig.java
Test Packages
Start Page
Alunos.java
Source
History
package com.is.db;
// Imports
import java.io.Serializable;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
@XmlRootElement
Warning
The target server has not been set or has been removed.
Right-click the project in the Projects window and choose
Resolve Missing Server Problem to set the target server.
OK

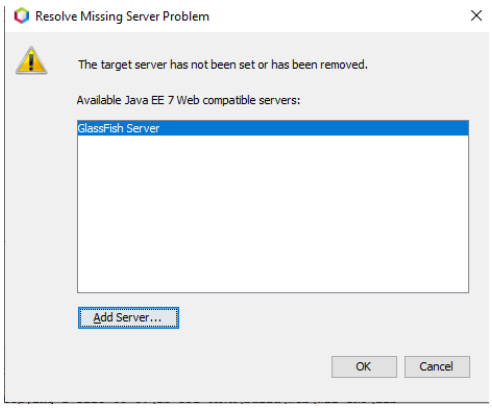

Resolve Missing Server Problem
The target server has not been set or has been removed.
The problem cannot be resolved because there is no suitable server instance at the moment. Please add a suitable server instance in the Servers manager.
Available Java EE 7 Web compatible servers:
Add Server...
OK
Cancel


Add Server Instance
Steps
1. Choose Server
2. ...
Choose Server
Server:
Amazon Beanstalk
Apache Tomcat or TomEE
GlassFish Server
Payara Server
WildFly Application Server
Name: GlassFish Server
< Back
Next >
Finish
Cancel
Help

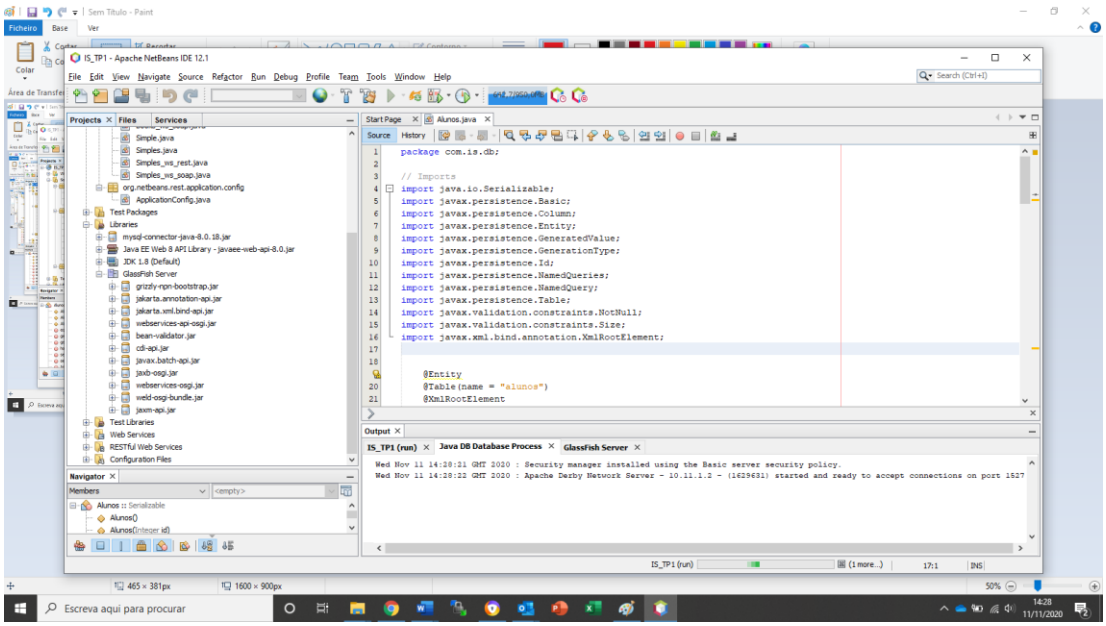

Add Server Instance
Steps
1. Choose Server
2. Server Location
3. Domain Name/Location
Server Location
Installation Location:
C:\Users\JMR\GlassFish_Server
Browse...
Local Domain
Remote Domain
Choose server to download:
GlassFish Server 5.1.0
Download Now...
I have read and accept the license agreement... (click)
C:\Users\JMR\GlassFish_Server does not exist but server can be downloaded and installed there.
< Back
Next >
Finish
Cancel
Help


Add Server Instance
Steps
1. Choose Server
2. Server Location
3. Domain Name/Location
Server Location
Installation Location:
C:\Users\JMR\GlassFish_Server
Browse...
Local Domain
Remote Domain
Choose server to download:
GlassFish Server 5.1.0
Cancel Download
I have read and accept the license agreement... (click)
14.8m of 111.9m : Installing glassfish/modules/ant.jar
< Back
Next >
Finish
Cancel
Help

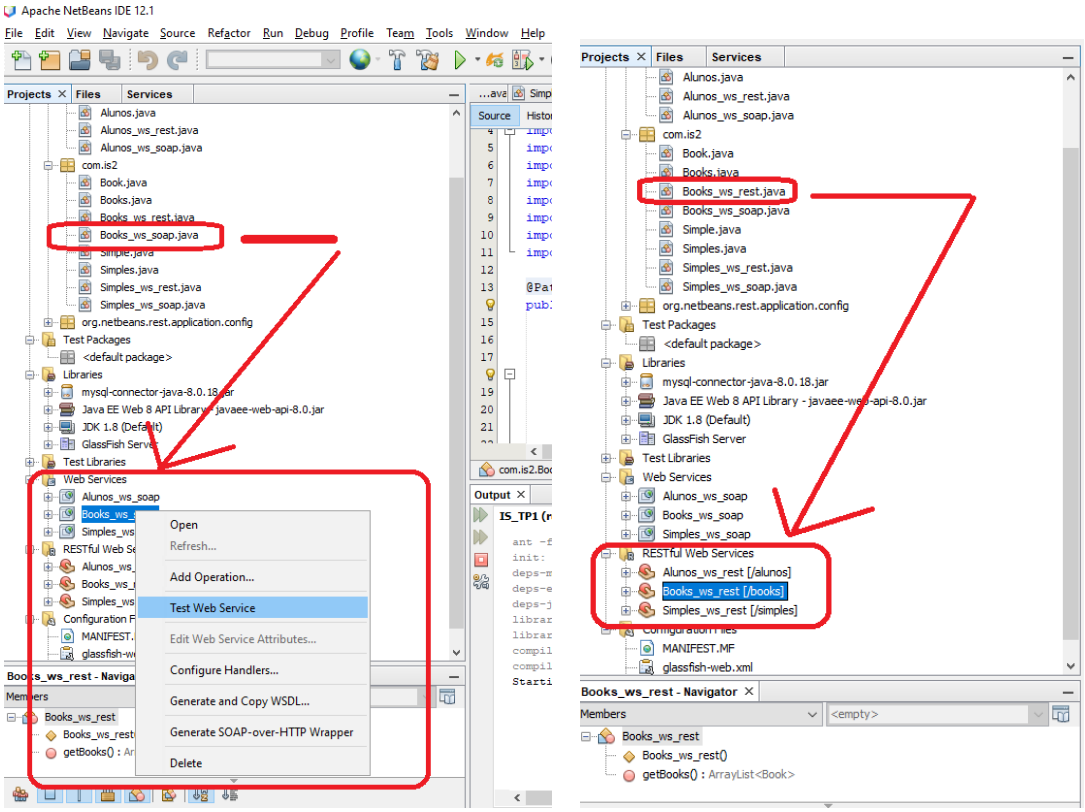

Add Server Instance
Steps
1. Choose Server
2. Server Location
3. Domain Name/Location
Server Location
Installation Location:
C:\Users\JMR\GlassFish_Server
Browse...
Local Domain
Remote Domain
Choose server to download:
GlassFish Server 5.1.0
Download Complete
I have read and accept the license agreement... (click)
Download & Install completed in 22 seconds.
Detected a GlassFish Server 5.1.0 install. Click Next to register remote or custom local domains.
< Back
Next >
Finish
Cancel
Help
Add Server Instance
Steps
1. Choose Server
2. Server Location
3. Domain Name/Location
Domain Location
Domain: domain1
Host: localhost
Loopback
DAS Port: 4848
HTTP Port: 8080
Default
Target:
User Name:
Password:
Register existing embedded domain: domain1
< Back
Next >
Finish
Cancel
Help



O projeto está pronto para ser executado:



Neste projeto o URL do tipo de webservices são diferentes:

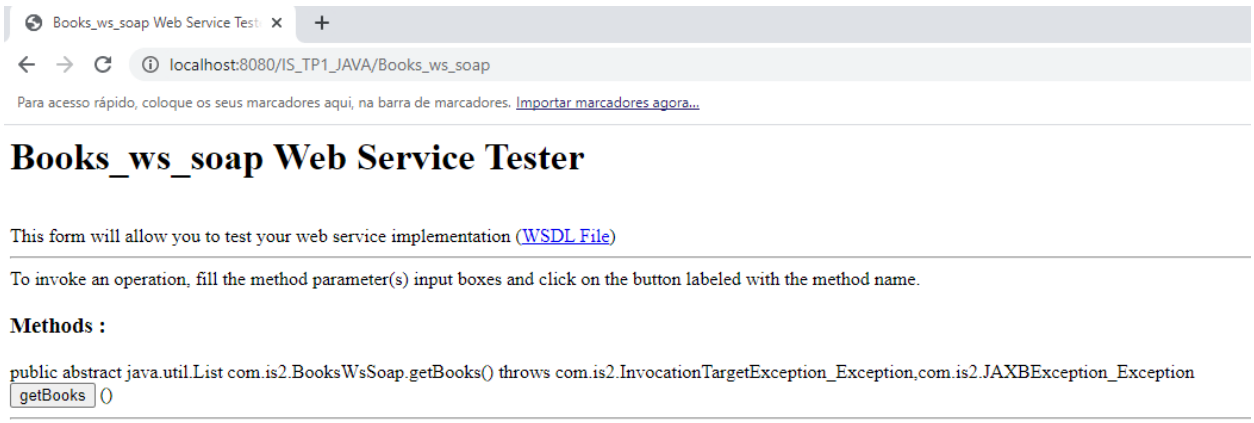


Using SOAP: http://localhost:8080/IS_TP1_JAVA/Books_ws_soap?Tester

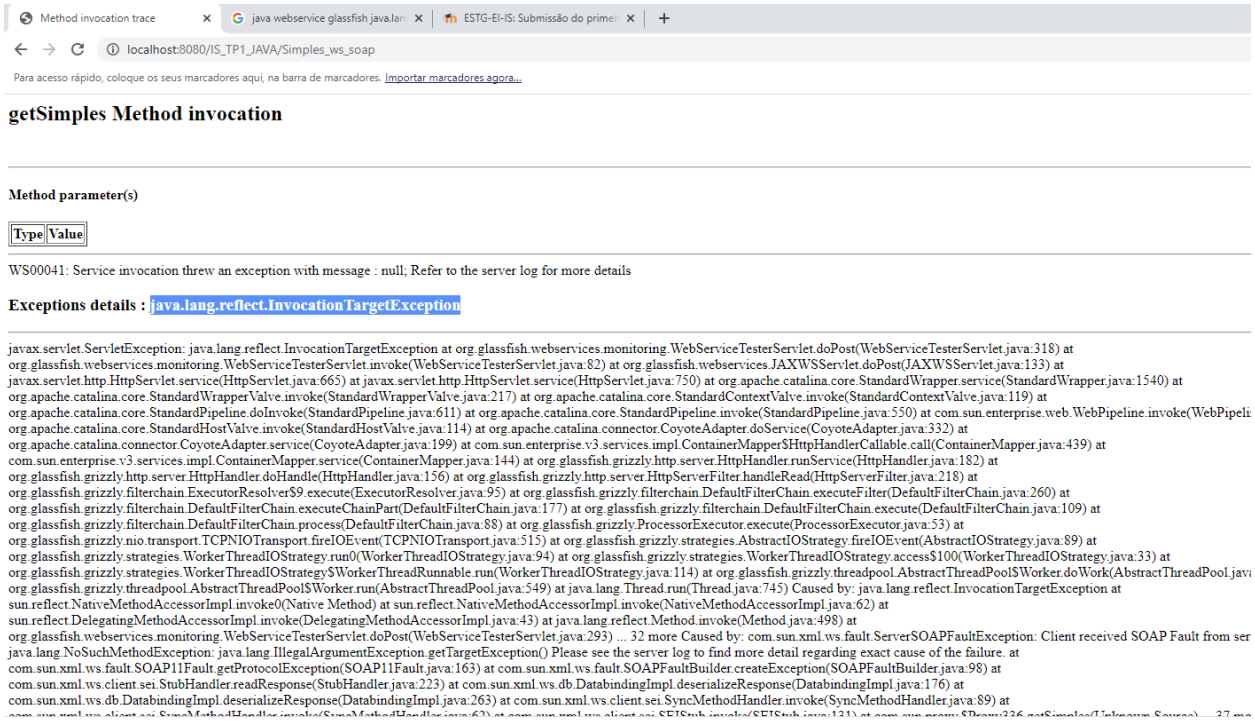
Using REST: http://localhost:8080/IS_TP1_JAVA/webresources/books

Testando o webservice: http://localhost:8080/IS_TP1_JAVA/Simples_ws_soap**?Tester** **ou**
http://localhost:8080/IS_TP1_JAVA/Books_ws_soap**?Tester**

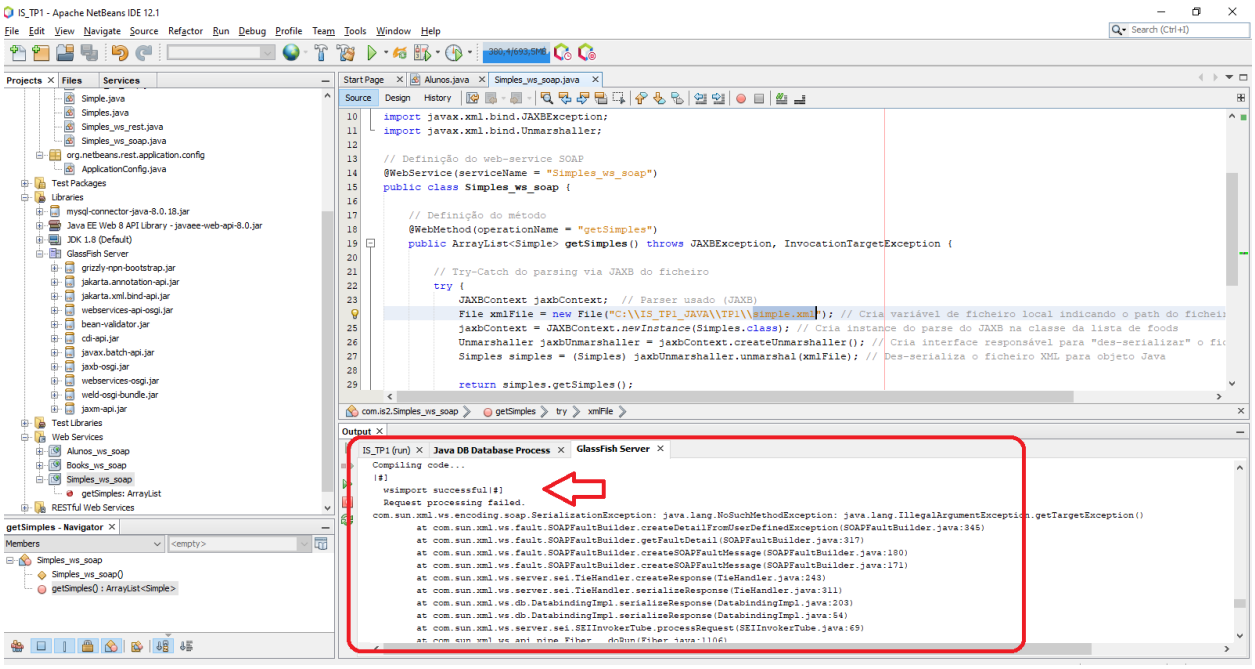
NOTA: Indicar no URL a expressão ?Tester



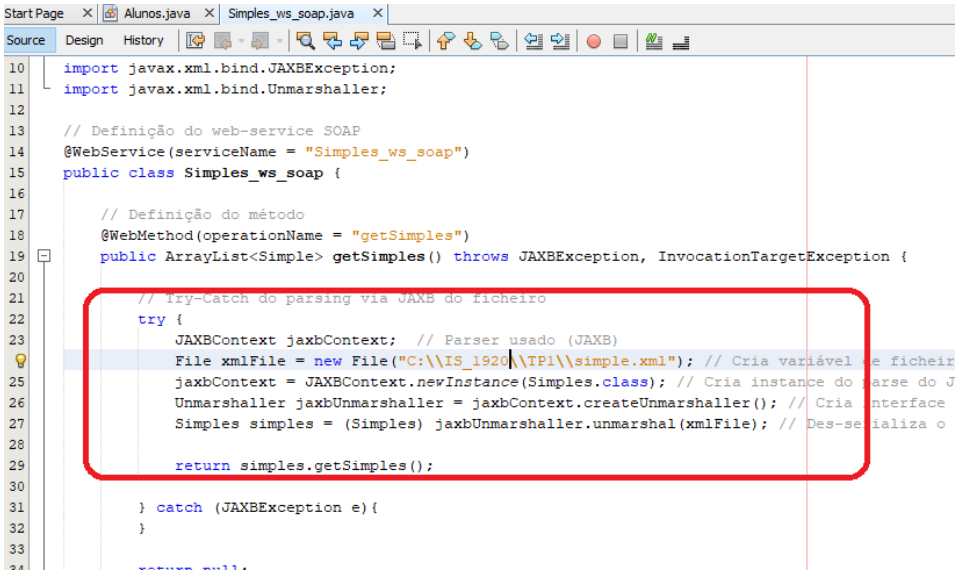
Surgirá uma excepção:



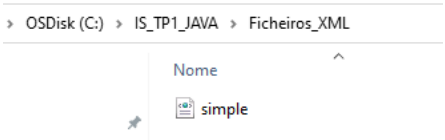
Erro apresentado no ambiente de desenvolvimento:



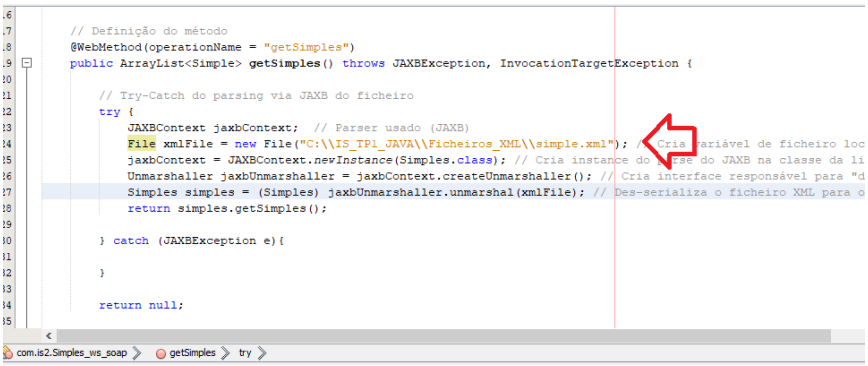
Resolução: Não consegue ler o ficheiro XML. Retificação:



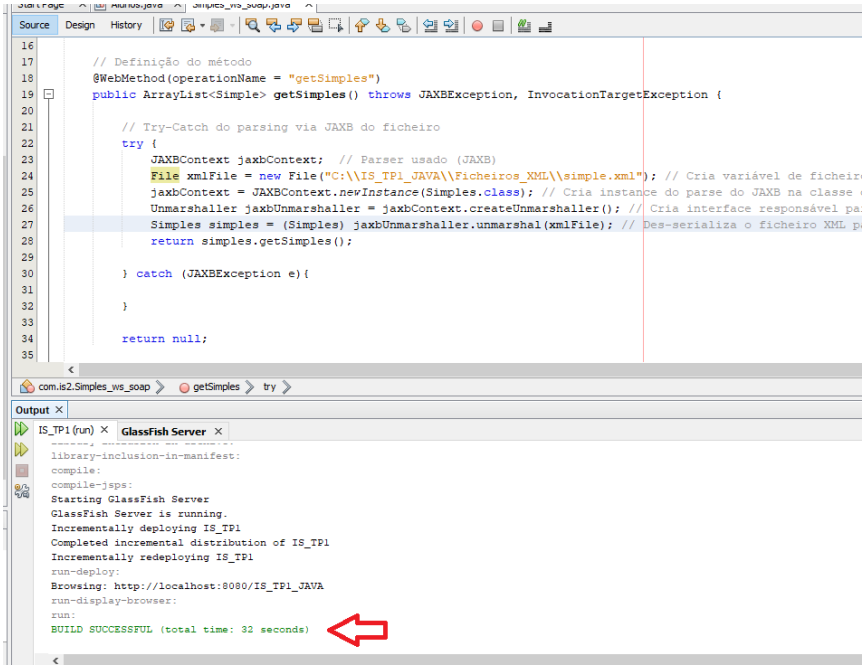
Criar uma pasta e copiar o ficheiro simple.xml para uma subpasta:



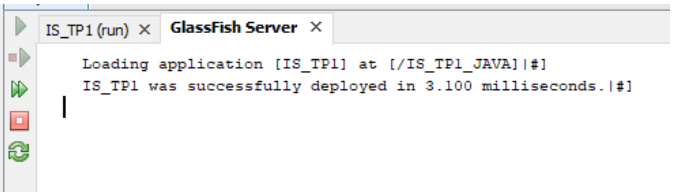
Retificar o no código:

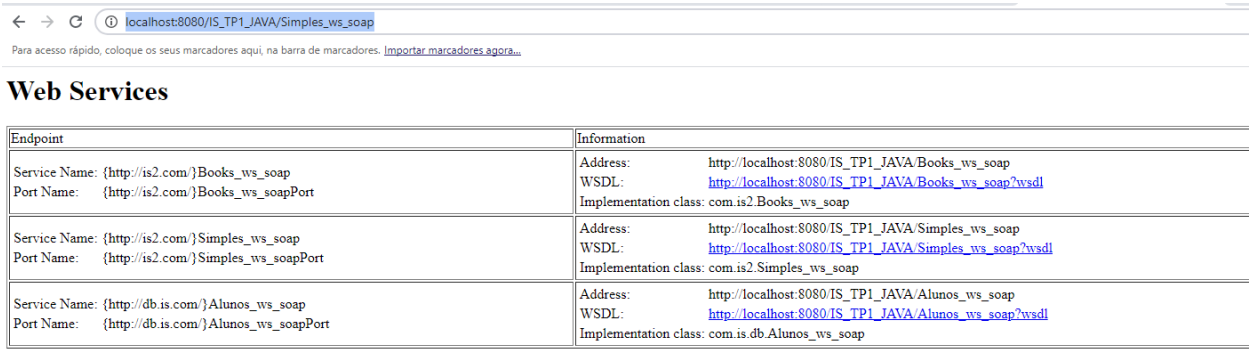


Fazer RUN:

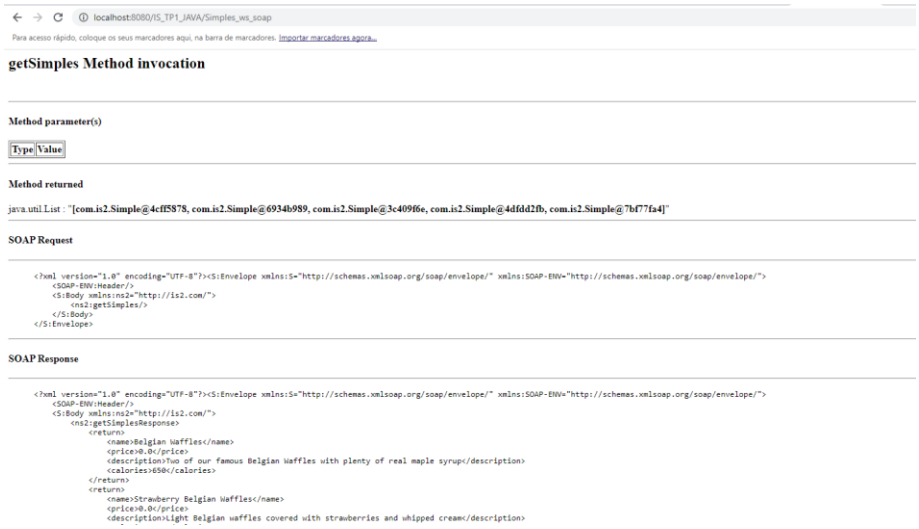


E os logs do servidor GlassFish:

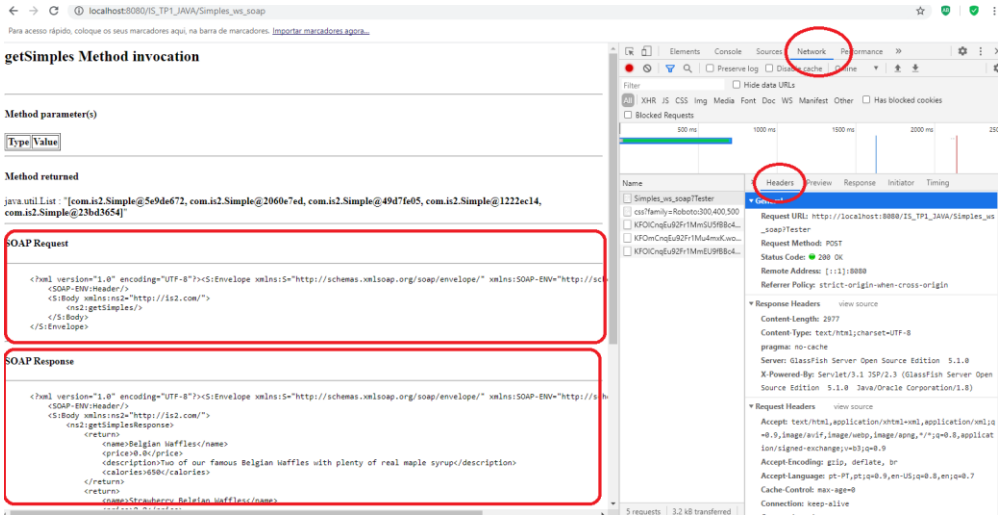


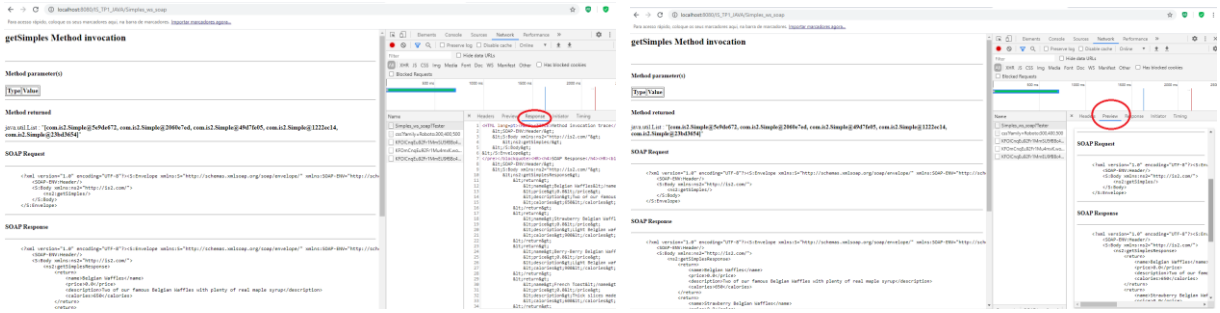


Ao executar o pedido no browser:



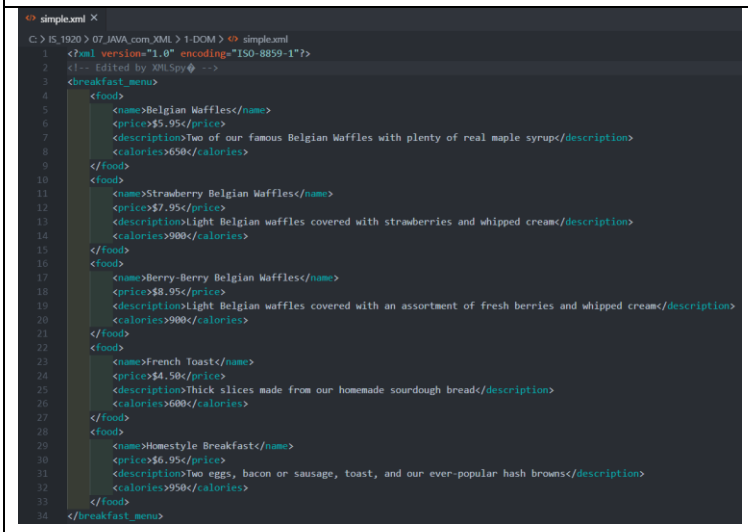
Analise a informação transferida entre o cliente e o servidor:





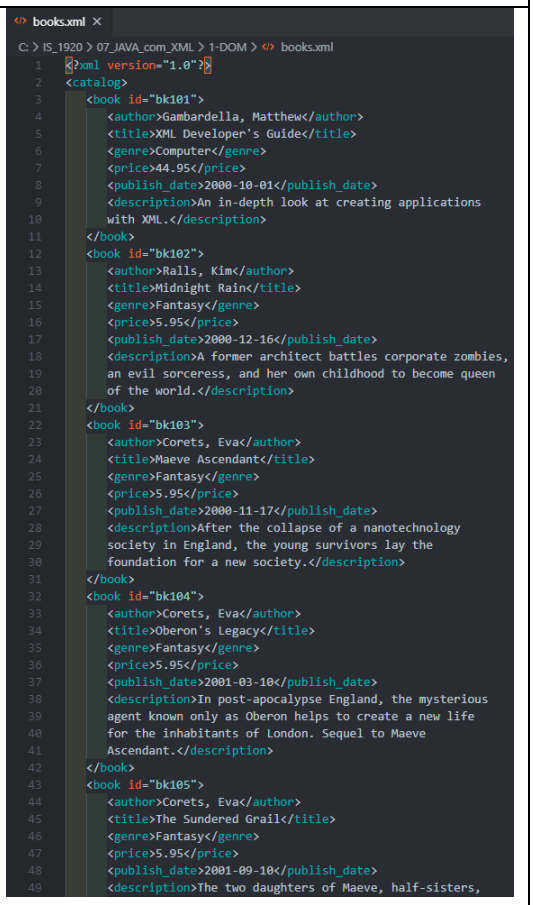
Os ficheiros manipulados através dos webservices serão os seguintes três:

simple.xml



```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Edited by XMLSpy® -->
<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>
</breakfast_menu>
```

books.xml

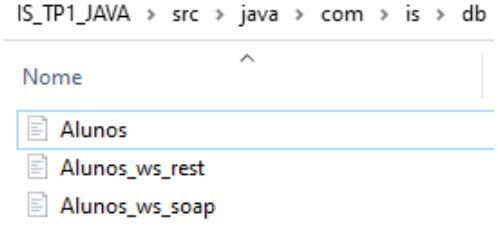


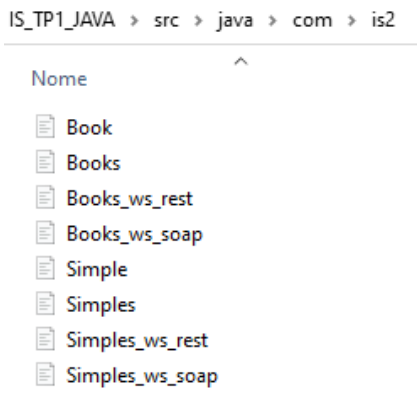
```xml
<?xml version="1.0"?>
<catalog>
    <book id="bk101">
        <author>Gambardella, Matthew</author>
        <title>XML Developer's Guide</title>
        <genre>Computer</genre>
        <price>44.95</price>
        <publish_date>2000-10-01</publish_date>
        <description>An in-depth look at 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 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,
```

alunos.xml:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<alunos>
    <aluno>
        <id>1</id>
        <nome>Antonio Silva</nome>
    </aluno>
    <aluno>
        <id>2</id>
        <nome>Maria Albertina</nome>
    </aluno>
    <aluno>
        <id>3</id>
        <nome>Jose Ferreira</nome>
    </aluno>
    <aluno>
        <id>4</id>
        <nome>Pedro Costa</nome>
    </aluno>
</alunos>
```

Em termos de estrutura dos ficheiros com o código fonte, os mesmos são disponibilizados nas diretorias “is” e “is2”, em que na primeira apresenta a classe “Alunos.java” e os Webservices em REST e em SOAP respetivamente:





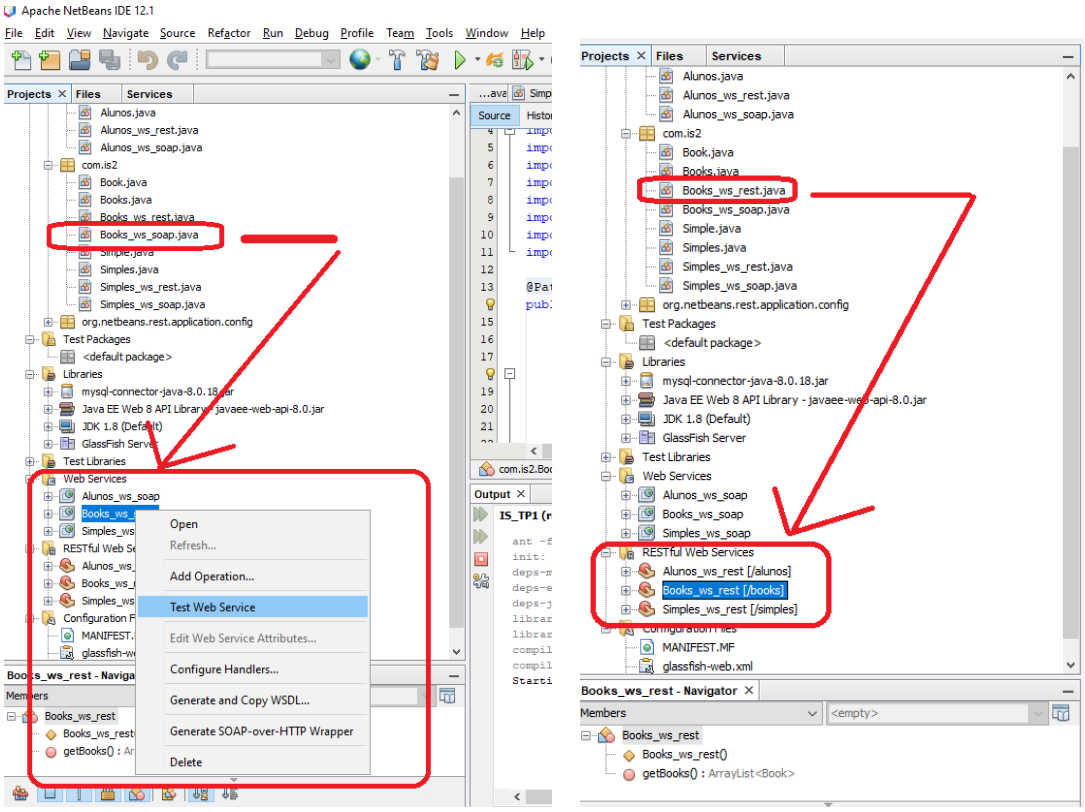
e

NOTA: As configurações de acesso à base de dados estão definidas no ficheiro “IS_TP1_JAVA\src\java\com\is\db\Alunos_ws_rest.java”:

Alunos_ws_soap.java

```java
package com.is.db;

// Imports
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import javax.jws.WebService;
import javax.jws.WebMethod;

// Definição do web-service SOAP
@WebService(serviceName = "Alunos_ws_soap")
public class Alunos_ws_soap {

    // Definição do método
    @WebMethod(operationName = "getAlunos")
    public ArrayList<Alunos> getAlunos() {
        ArrayList<Alunos> alunos = new ArrayList<>();

        // Try-Catch de ligação à BD e query da tabela alunos
        try{
            Class.forName("com.mysql.jdbc.Driver");
            String url = "jdbc:mysql://localhost:3306/is?useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimezone=UTC";
            Connection conn = DriverManager.getConnection(url, "userdb", "teste"); // Conexão com credenciais de ligação do MySQL
            Statement st = conn.createStatement();
            ResultSet srs = st.executeQuery("SELECT * FROM alunos");
```

Neste projeto, a implementação dos WebServices são realizados em links diferentes:



Em SOAP: http://localhost:8080/IS_TP1_JAVA/Books_ws_soap?Tester

Em REST: http://localhost:8080/IS_TP1_JAVA/webresources/books

2.2.1 Importação do ficheiro books.xml, via SOAP e Parser: JAXB

Para o carreamento da informação do ficheiro "books.xml" foram criadas as classes Book.java e Boks.java

Book.java

```java
package com.is2;

// Imports
import java.io.Serializable;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement(name = "book") // Tag raiz para cada livro
@XmlAccessorType(XmlAccessType.FIELD)
public class Book implements Serializable{
    // Atributos para cada tag / elemento
    private String id;
    private String author;
    private String title;
    private String genre;
    private float price;
    private String publish_date;
    private String description;

    // Getters & Setters
    public String getId() {...3 lines }

    public void setId(String id) {...3 lines }

    public String getAuthor() {...3 lines }

    public void setAuthor(String author) {...3 lines }

    public String getTitle() {...3 lines }

    public void setTitle(String title) {...3 lines }

    public String getGenre() {...3 lines }

    public void setGenre(String genre) {...3 lines }

    public float getPrice() {...3 lines }

    public void setPrice(float price) {...3 lines }

    public String getPublish_date() {...3 lines }

    public void setPublish_date(String publish_date) {...3 lines }

    public String getDescription() {...3 lines }
```

Books.java

```java
package com.is2;

// Imports
import java.util.ArrayList;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement(name = "catalog") // Tag raíz da lista da livros
@XmlAccessorType(XmlAccessType.FIELD)
public class Books {

    @XmlElement(name = "book")
    private ArrayList<Book> books = null; // Lista de elementos de tipo Book

    // Getters & Setters
    public ArrayList<Book> getBooks() {...3 lines }

    public void setBooks(ArrayList<Book> books) {...3 lines }

}
```

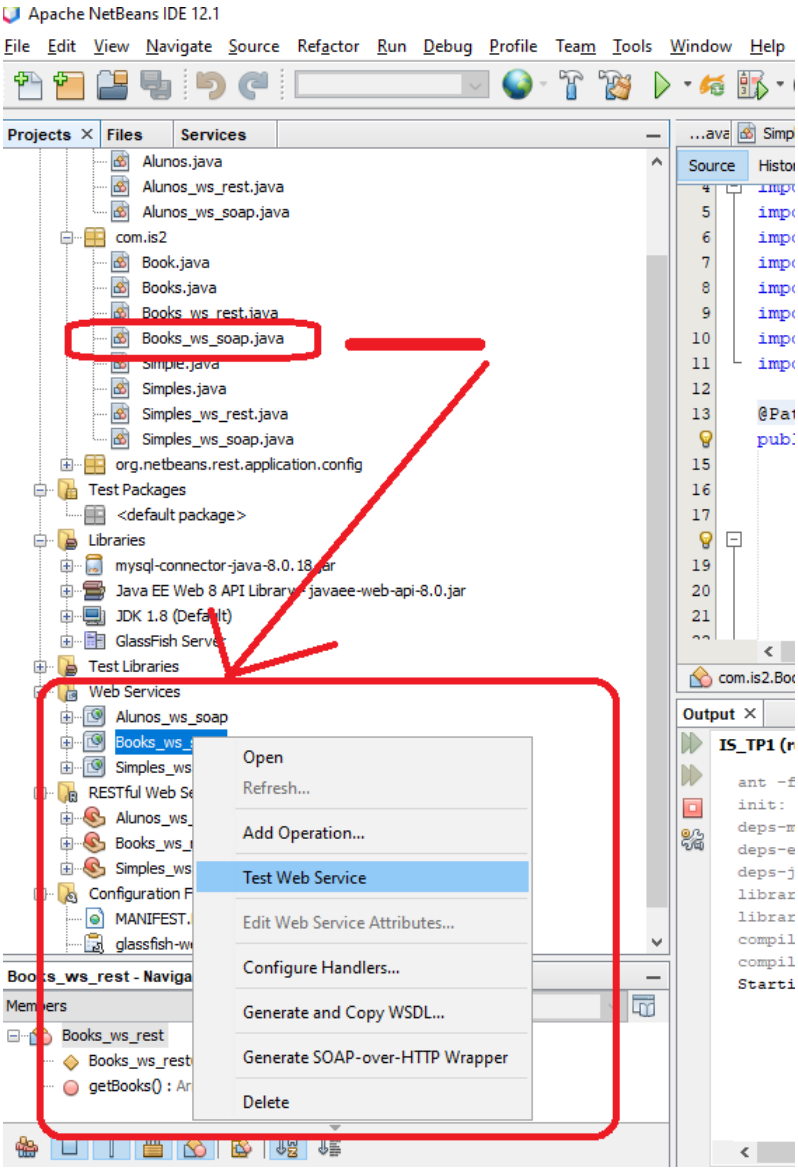
O Webservice usando SOAP e a abordagem JAXB é a seguinte:

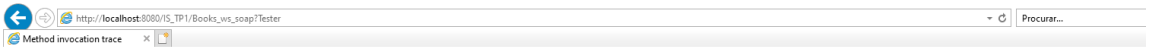
Books_ws_soap.java

```java
package com.is2;

// Imports
import java.io.File;
import java.lang.reflect.InvocationTargetException;
import java.util.ArrayList;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Unmarshaller;

// Definição do web-service SOAP
@WebService(serviceName = "Books_ws_soap")
public class Books_ws_soap {

    // Definição do método
    @WebMethod(operationName = "getBooks")
    public ArrayList<Book> getBooks() throws JAXBException, InvocationTargetException {

        // Try-Catch do parsing via JAXB do ficheiro
        try {
            JAXBContext jaxbContext;  // Parser usado (JAXB)
            File xmlFile = new File("C:\\IS_1920\\TP1\\books.xml"); // Cria variável de ficheiro local indicando o path do ficheiro "books.xml"
            jaxbContext = JAXBContext.newInstance(Books.class); // Cria instance do parse do JAXB na classe da lista de livros
            Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller(); // Cria interface responsável para "des-serializar" o ficheiro XML para objeto Java
            Books books = (Books) jaxbUnmarshaller.unmarshal(xmlFile); // Des-serializa o ficheiro XML para objeto Java

            return books.getBooks();

        } catch (JAXBException e){
        }

        return null;

    }
}
```

O Teste do webservice é o seguinte:

A partir do NetBeans, execute o webservice:





SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="ht
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:getBooks xmlns:ns2="http://is2.com/"/>
    </S:Body>
</S:Envelope>
```

SOAP Response

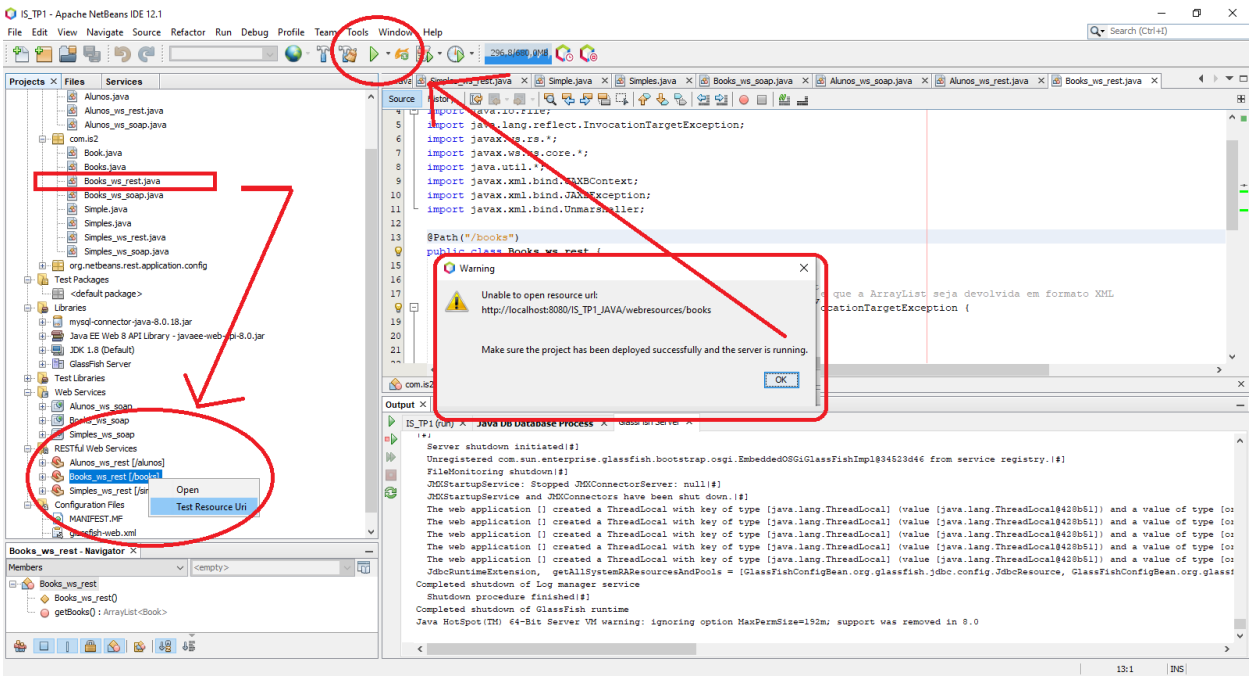
```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="ht
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:getBooksResponse xmlns:ns2="http://is2.com/">
            <return>
                <author>Gambardella, Matthew</author>
                <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>
            </return>
            <return>
                <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>
            </return>
            <return>
```

2.2.2 Importação do ficheiro books.xml, via REST e Parser: JAXB

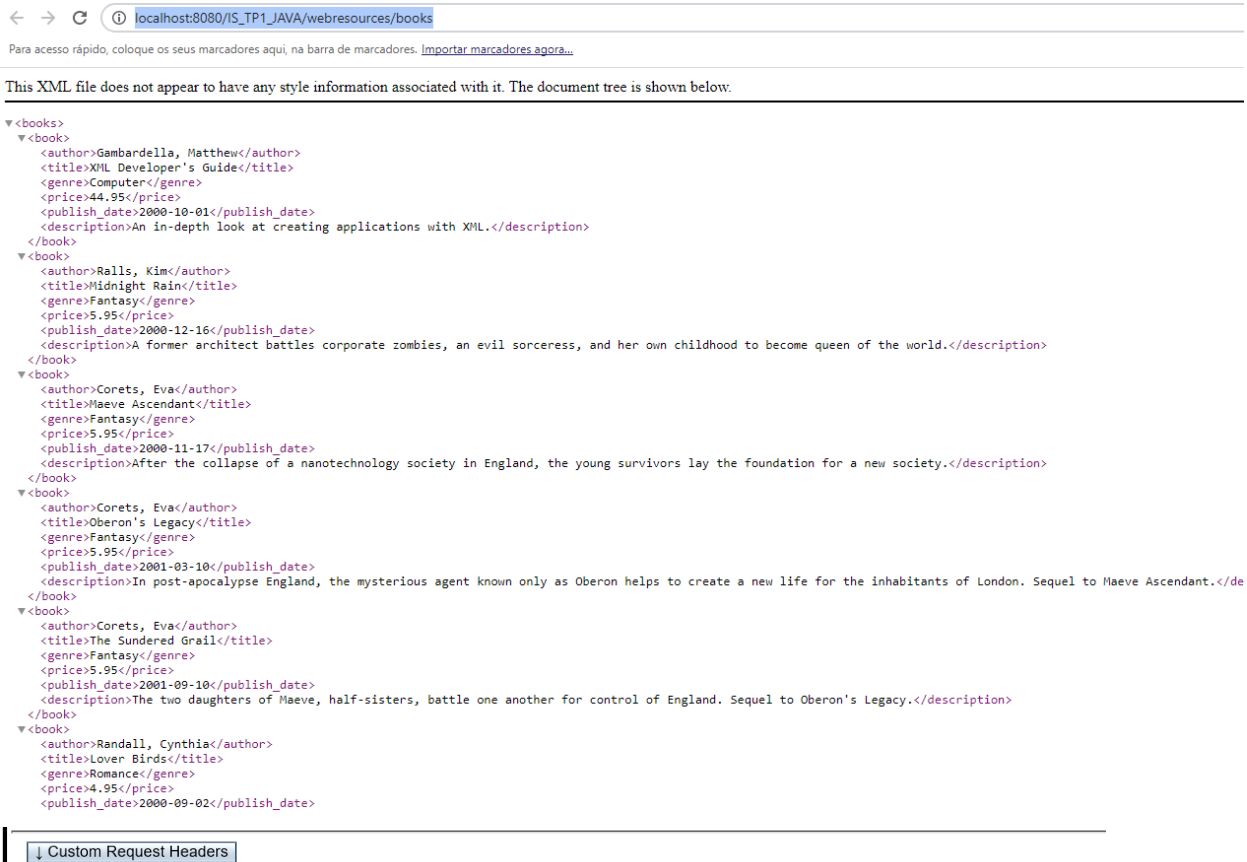
O Webservice usando REST e a abordagem JAXB é a seguinte:

```java
package com.is2;

// Imports
import java.io.File;
import java.lang.reflect.InvocationTargetException;
import javax.ws.rs.*;
import javax.ws.rs.core.*;
import java.util.*;
import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Unmarshaller;

@Path("/books")
public class Books_ws_rest {

    @GET
    @Produces(MediaType.APPLICATION_XML) // Anotação que permite que a ArrayList seja devolvida em formato XML
    public ArrayList<Book> getBooks() throws JAXBException, InvocationTargetException {

        // Try-Catch do parsing via JAXB do ficheiro
        try {
            JAXBContext jaxbContext; // Parser usado (JAXB)
            File xmlFile = new File("C:\\IS_1920\\TP1\\books.xml"); // Cria variável de ficheiro local indicando o path do ficheiro "books.xml"
            jaxbContext = JAXBContext.newInstance(Books.class); // Cria instance do parse do JAXB na classe da lista de livros
            Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller(); // Cria interface responsável para "des-serializar" o ficheiro XML para objeto Java
            Books books = (Books) jaxbUnmarshaller.unmarshal(xmlFile); // Des-serializa o ficheiro XML para objeto Java

            return books.getBooks();

        }catch (JAXBException e) {
        }

        return null;

    }
}
```

Para testar o Webservice em REST:



O Teste do webservice é o seguinte: http://localhost:8080/IS_TP1_JAVA/webresources/books


localhost:8080/IS_TP1_JAVA/webresources/books
Para acesso rápido, coloque os seus marcadores aqui, na barra de marcadores. Importar marcadores agora...
This XML file does not appear to have any style information associated with it. The document tree is shown below.
<books>
<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.</de
</book>
<book>
<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>
<author>Randall, Cynthia</author>
<title>Lover Birds</title>
<genre>Romance</genre>
<price>4.95</price>
<publish_date>2000-09-02</publish_date>
↓ Custom Request Headers
Status: 200 (OK)
Response:
Tabular View
Raw View
Sub-Resource
Headers
Http Monitor
<?xml version="1.0" encoding="UTF-8"?>
<books>
<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>

2.2.3 Importação do ficheiro simple.xml, via SOAP e Parser: JAXB

Para o carreamento da informação do ficheiro "simple.xml" foram criadas as classes Simple.java e Simples.java

Simple.java

```java
package com.is2;

// Imports
import java.io.Serializable;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement(name = "food") // Tag raiz para cada elemento
@XmlAccessorType(XmlAccessType.FIELD)
public class Simple implements Serializable{
    // Atributos para cada tag / elemento
    private String name;
    private float price;
    private String description;
    private int calories;

    // Getters & Setters
    public String getName() {...3 lines }

    public void setName(String name) {...3 lines }

    public float getPrice() {...3 lines }

    public void setPrice(float price) {...3 lines }

    public String getDescription() {...3 lines }

    public void setDescription(String description) {...3 lines }

    public int getCalories() {...3 lines }

    public void setCalories(int calories) {...3 lines }

    }
```

Simples.java

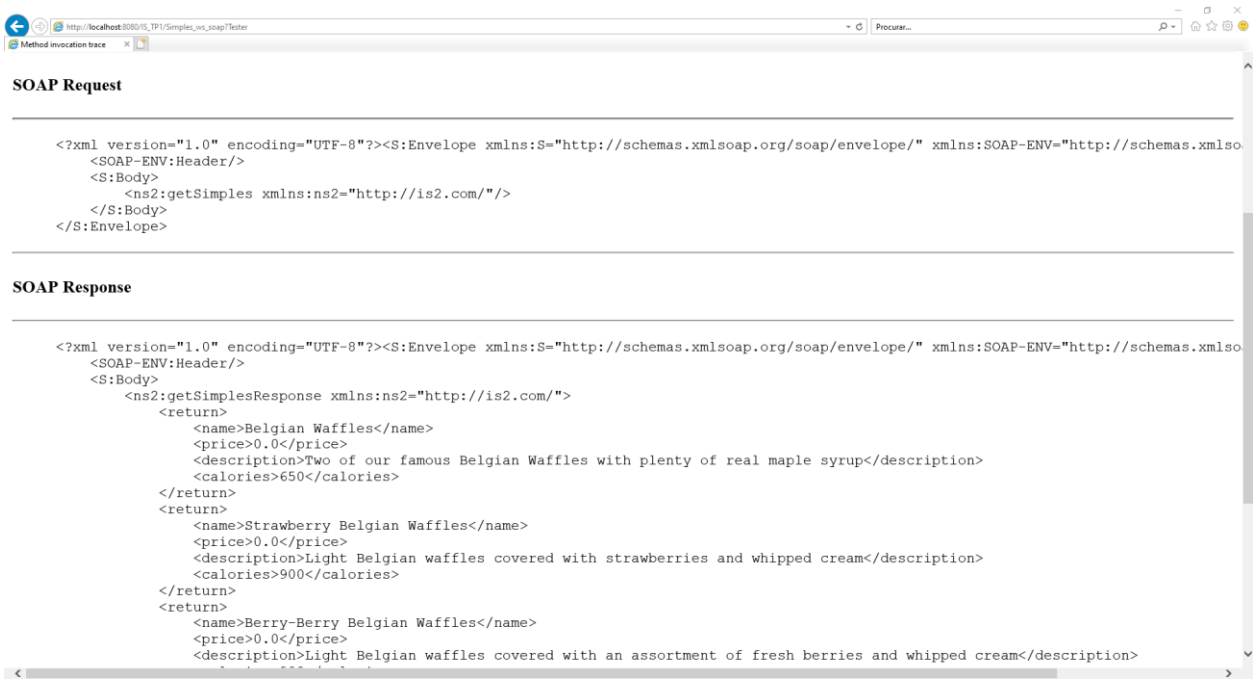
```java
package com.is2;

// Imports
import java.util.ArrayList;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement(name = "breakfast_menu") // Tag raíz da lista de foods
@XmlAccessorType(XmlAccessType.FIELD)
public class Simples {

  @XmlElement(name = "food")
    private ArrayList<Simple> simples = null; // Lista de elementos de tipo food

    // Getters & Setters
    public ArrayList<Simple> getSimples() {...3 lines }

    public void setSimples(ArrayList<Simple> simples) {...3 lines }

}
```

O webservice usando SOAP e a abordagem JAXB é a seguinte:

Simples_ws_soap.java

```java
package com.is2;

// Imports
import java.io.File;
import java.lang.reflect.InvocationTargetException;
import java.util.ArrayList;
import javax.jws.WebService;
import javax.jws.WebMethod;
import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Unmarshaller;

// Definição do web-service SOAP
@WebService(serviceName = "Simples_ws_soap")
public class Simples_ws_soap {

    // Definição do método
    @WebMethod(operationName = "getSimples")
    public ArrayList<Simple> getSimples() throws JAXBException, InvocationTargetException {

        // Try-Catch do parsing via JAXB do ficheiro
        try {
            JAXBContext jaxbContext;  // Parser usado (JAXB)
            File xmlFile = new File("C:\\IS_1920\\TP1\\simple.xml"); // Cria variável de ficheiro local indicando o path do ficheiro "'simple.xml"
            jaxbContext = JAXBContext.newInstance(Simples.class); // Cria instance do parse do JAXB na classe da lista de foods
            Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller(); // Cria interface responsável para "des-serializar" o ficheiro XML para objeto Java
            Simples simples = (Simples) jaxbUnmarshaller.unmarshal(xmlFile); // Des-serializa o ficheiro XML para objeto Java

            return simples.getSimples();

        } catch (JAXBException e){
        }

        return null;

    }
}
```

O teste do Webservice é o seguinte:



SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://schemas.xmlso
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:getSimples xmlns:ns2="http://is2.com/"/>
    </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.xmlso
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:getSimplesResponse xmlns:ns2="http://is2.com/">
            <return>
                <name>Belgian Waffles</name>
                <price>0.0</price>
                <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
                <calories>650</calories>
            </return>
            <return>
                <name>Strawberry Belgian Waffles</name>
                <price>0.0</price>
                <description>Light Belgian waffles covered with strawberries and whipped cream</description>
                <calories>900</calories>
            </return>
            <return>
                <name>Berry-Berry Belgian Waffles</name>
                <price>0.0</price>
                <description>Light Belgian waffles covered with an assortment of fresh berries and whipped cream</description>
```

2.2.4 Importação do ficheiro simple.xml, via REST e Parser: JAXB

A definição do webservice em REST é o seguinte:

Simples_ws_rest.java

```
import java.io.File;
import java.lang.reflect.InvocationTargetException;
import javax.ws.rs.*;
import javax.ws.rs.core.*;
import java.util.*;
import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Unmarshaller;

@Path("/simples")
public class Simples_ws_rest {

    @GET
    @Produces(MediaType.APPLICATION_XML) // Anotação que permite que a ArrayList seja devolvida em formato XML
    public ArrayList<Simple> getSimples() throws JAXBException, InvocationTargetException {

        // Try-Catch do parsing via JAXB do ficheiro
        try {
            JAXBContext jaxbContext; // Parser usado (JAXB)
            File xmlFile = new File("C:\\IS_1920\\TP1\\simple.xml"); // Cria variável de ficheiro local indicando o path do ficheiro ""simple.xml"
            jaxbContext = JAXBContext.newInstance(Simples.class); // Cria instance do parse do JAXB na classe da lista de livros
            Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller(); // Cria interface responsável para "des-serializar" o ficheiro XML para objeto Java
            Simples simples = (Simples) jaxbUnmarshaller.unmarshal(xmlFile); // Des-serializa o ficheiro XML para objeto Java

            return simples.getSimples();

        }catch (JAXBException e) {
        }

        return null;


    }

}
```

O teste do webservice é o seguinte:

↓ Custom Request Headers

Status: 200 (OK)

Response:

Tabular View | Raw View | Sub-Resource | Headers | Http Monitor

```
<?xml version="1.0" encoding="UTF-8"?>
  <simples>
    <food>
      <name>Belgian Waffles</name>
      <price>0.0</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>0.0</price>
      <description>Light Belgian waffles covered with strawberries and whipped cream</description>
      <calories>900</calories>
    </food>
    <food>
      <name>Berry-Berry Belgian Waffles</name>
      <price>0.0</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>
```

2.2.5 Importação do ficheiro alunos.xml, via SOAP e importação na Base de Dados

Considere-se o ficheiro alunos.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
- <alunos>
  - <aluno>
      <id>1</id>
      <nome>Antonio Silva</nome>
    </aluno>
  - <aluno>
      <id>2</id>
      <nome>Maria Albertina</nome>
    </aluno>
  - <aluno>
      <id>3</id>
      <nome>Jose Ferreira</nome>
    </aluno>
  - <aluno>
      <id>4</id>
      <nome>Pedro Costa</nome>
    </aluno>
</alunos>
```

Criou-se a classe a Classe Alunos.java:

Alunos.java

```java
package com.is.db;

// Imports
import java.io.Serializable;
import javax.persistence.Basic;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import javax.xml.bind.annotation.XmlRootElement;


    @Entity
    @Table(name = "alunos")
    @XmlRootElement
    // Definição de métodos equiparados a queries
    @NamedQueries({
    @NamedQuery(name = "Alunos.findAll", query = "SELECT a FROM Alunos a"),
    @NamedQuery(name = "Alunos.findById", query = "SELECT a FROM Alunos a WHERE a.id = :id"),
    @NamedQuery(name = "Alunos.findByNome", query = "SELECT a FROM Alunos a WHERE a.nome = :nome")})
    public class Alunos implements Serializable {

    // Atributos
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id")
    private Integer id;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 200)
    @Column(name = "nome")
    private String nome;

```

O webservice implementado em SOAP é o seguinte:

Alunos_ws_soap.java

```java
package com.is.db;

// Imports
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import javax.jws.WebService;
import javax.jws.WebMethod;

// Definição do web-service SOAP
@WebService(serviceName = "Alunos_ws_soap")
public class Alunos_ws_soap {

    // Definição do método
    @WebMethod(operationName = "getAlunos")
    public ArrayList<Alunos> getAlunos() {
        ArrayList<Alunos> alunos = new ArrayList<>();

        // Try-Catch de ligação à BD e query da tabela alunos
        try{
          Class.forName("com.mysql.jdbc.Driver");
          String url = "jdbc:mysql://localhost:3306/is?useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimezone=UTC";
          Connection conn = DriverManager.getConnection(url, "root", "root"); // Conexão com credenciais de ligação do MySQL
          Statement st = conn.createStatement();
          ResultSet srs = st.executeQuery("SELECT * FROM alunos");

          // Adição do resultado da query à resposta XML
          while (srs.next()) {
              Alunos al = new Alunos();
              al.setId(srs.getInt("id"));
              al.setNome(srs.getString("nome"));
              alunos.add(al);
           }

          return alunos;

        }catch(ClassNotFoundException | SQLException e){
        }

        return null;
    }
}

```

A execução do webservice é a seguinte:

SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?><S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENV="http://s
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:getAlunos xmlns:ns2="http://db.is.com/"/>
    </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://s
    <SOAP-ENV:Header/>
    <S:Body>
        <ns2:getAlunosResponse xmlns:ns2="http://db.is.com/">
            <return>
                <id>1</id>
                <nome> Antonio Silva </nome>
            </return>
            <return>
                <id>1</id>
                <nome> Maria Albertina </nome>
            </return>
            ...

        </ns2:getAlunosResponse>
    </S:Body>
</S:Envelope>
```

2.2.6 Importação do ficheiro alunos.xml, via REST e importação na Base de Dados

Alunos_ws_rest.java

```java
package com.is.db;

// Imports
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import javax.ws.rs.*;
import javax.ws.rs.core.*;
import java.util.*;

@Path("/alunos")
public class Alunos_ws_rest {

    @GET
    @Produces(MediaType.APPLICATION_XML) // Anotação que permite que a ArrayList seja devolvida em formato XML
    public ArrayList<Alunos> getAlunos() {
        ArrayList<Alunos> alunos = new ArrayList<>();

        // Try-Catch de ligação à BD e query da tabela alunos
        try{
            Class.forName("com.mysql.jdbc.Driver");
            String url = "jdbc:mysql://localhost:3306/is?useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimezone=UTC";
            Connection conn = DriverManager.getConnection(url, "root", "root"); // Conexão com credenciais de ligação do MySQL
            Statement st = conn.createStatement();
            ResultSet srs = st.executeQuery("SELECT * FROM alunos");

            // Adição do resultado da query à resposta XML
            while (srs.next()) {
                Alunos al = new Alunos();
                al.setId(srs.getInt("id"));
                al.setNome(srs.getString("nome"));
                alunos.add(al);
            }

            return alunos;

        }catch(ClassNotFoundException | SQLException e){
        }

        return null;
    }

}
```

A execução do webservice é a seguinte:

↓ Custom Request Headers

Status: 200 (OK)

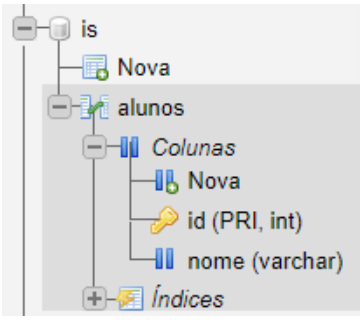
Response:

Tabular View | Raw View | Sub-Resource | Headers | Http Monitor

```
<?xml version="1.0" encoding="utf-8"?>
<alunos>
  <aluno>
    <id>1</id>
    <nome>Antonio Silva</nome>
  </aluno>
  <aluno>
    <id>2</id>
    <nome>Maria Albertina</nome>
  </aluno>
  <aluno>
    <id>3</id>
    <nome>Jose Ferreira</nome>
  </aluno>
  <aluno>
    <id>4</id>
    <nome>Pedro Costa</nome>
  </aluno>
</alunos>
```

2.3 Implementação em PHP

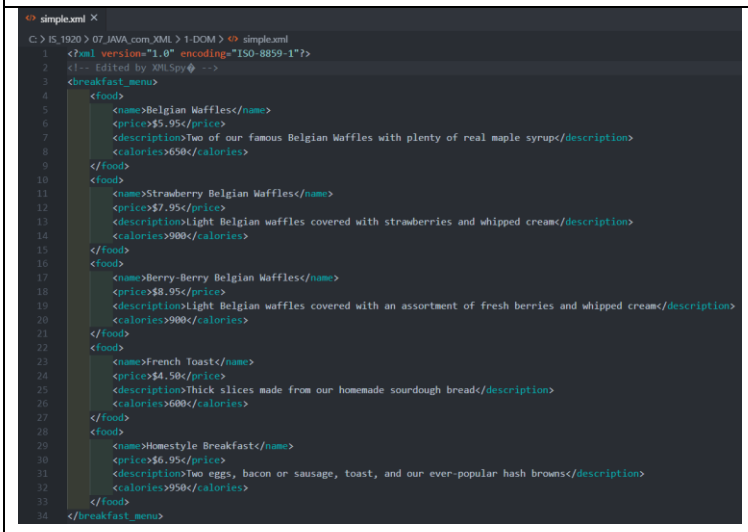
NOTA: No sentido de criar uma tabela na base de dados mySQL, crie uma base de dados com a designação “is” (diminutivo da designação da unidade curricular de Integração de sistemas)



	#	Nome	Tipo	Agrupamento (Collation)	Atributos	Nulo	Predefinido	Comentários	Extra	Acções
☐	1	id	int(11)			Não	*Nenhum*			Muda Elimina Mais
☐	2	nome	varchar(50)	utf8mb4_general_ci		Não	*Nenhum*			Muda Elimina Mais

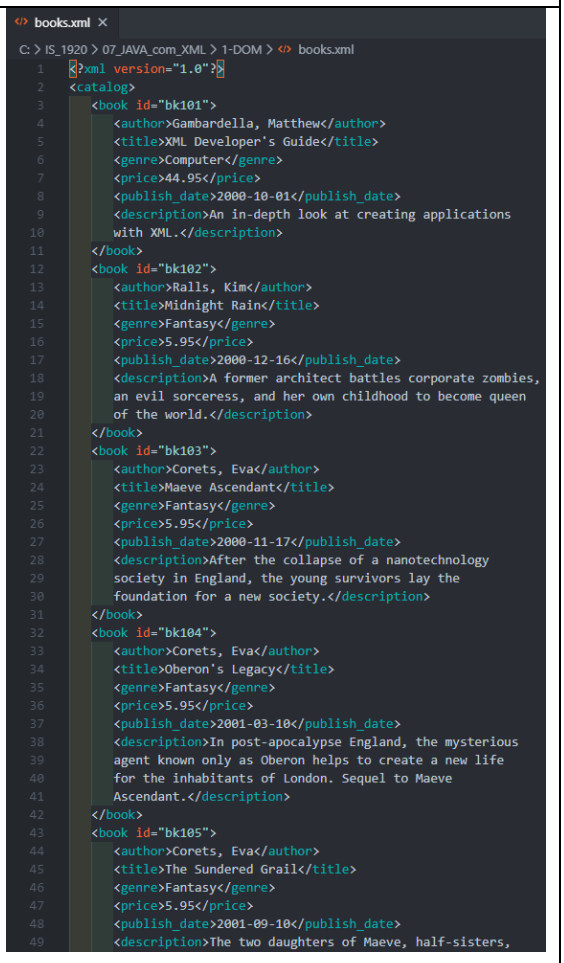
Os ficheiros manipulados através dos webservices serão os seguintes três:

simple.xml



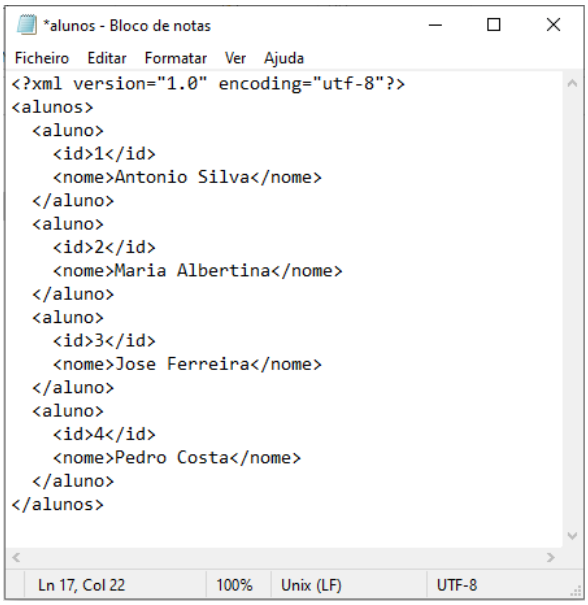
alunos.xml:

books.xml



```
<?xml version="1.0" encoding="UTF-8"?>
- <alunos>
  - <aluno>
      <id>1</id>
      <nome>Antonio Silva</nome>
    </aluno>
  - <aluno>
      <id>2</id>
      <nome>Maria Albertina</nome>
    </aluno>
  - <aluno>
      <id>3</id>
      <nome>Jose Ferreira</nome>
    </aluno>
  - <aluno>
      <id>4</id>
      <nome>Pedro Costa</nome>
    </aluno>
</alunos>
```

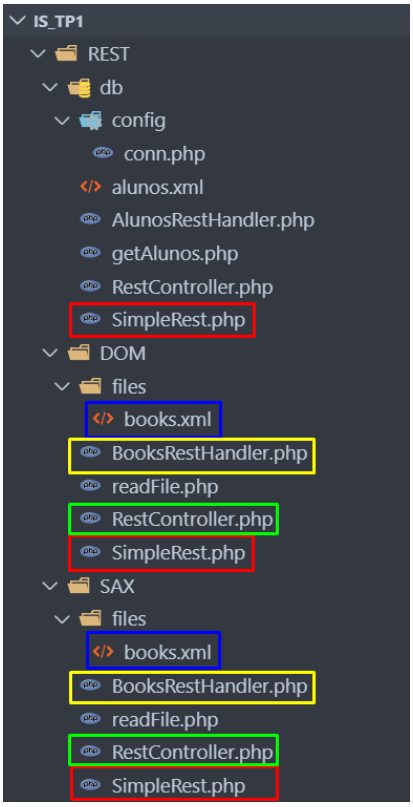
Para verificar se a importação do ficheiro XML está a ser importado corretamente, considere o ficheiro "alunos.xml"




```
<?xml version="1.0" encoding="UTF-8"?>
- <alunos>
  - <aluno>
      <id>1</id>
      <nome>Antonio Silva</nome>
    </aluno>
  - <aluno>
      <id>2</id>
      <nome>Maria Albertina</nome>
    </aluno>
  - <aluno>
      <id>3</id>
      <nome>Jose Ferreira</nome>
    </aluno>
  - <aluno>
      <id>4</id>
      <nome>Pedro Costa</nome>
    </aluno>
</alunos>
```

2.3.1 Execução de webservice via REST para leitura de dados de uma base de dados

Considere a diretoria “REST” em que os ficheiros com a mesma cor são iguais:



Considere também as configurações de acesso à base de dados:

```php
<?php
    $servername = "localhost:3306";
    $username = "userdb";
    $password = "teste";
    $database = "is";

    try {
        $PDO = new PDO("mysql:host=$servername;dbname=$database", $username, $password);

        $PDO->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $PDO->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);

    } catch (PDOException $e) {
        var_dump($e);
    }
?>
```

A implementação do Webservice poderá ser ilustrada através do ficheiro “AlunoRestHandler.php”:

AlunosRestHandler.php

REST > db > AlunosRestHandler.php

```php
<?php
require_once("SimpleRest.php");
require_once("getAlunos.php");

class AlunosRestHandler extends SimpleRest {

	function getAllAlunos() {

	$rawData = getAlunos();
		if(empty($rawData)) {
			$statusCode = 404;
			//$rawData = array('error' => 'No alunos found!');
		} else {
			$statusCode = 200;
		}

		$requestContentType = $_SERVER['HTTP_ACCEPT'];
		$this ->setHttpHeaders($requestContentType, $statusCode);

		if(strpos($requestContentType,'text/html') !== false){
		//print_r($rawData);
			header('Content-Type: application/xml');
			$response = $this->encodeXml($rawData);
			return $response;
		}
		}

	public function encodeXml($responseData) {
	$dom = new DOMDocument();
	$dom->encoding = 'utf-8';
	$dom->xmlVersion = '1.0';
	$dom->formatOutput = true;
	$xml_file_name = 'alunos.xml';
	$root = $dom->createElement('alunos');

	foreach ($responseData as $row) {
		$aluno_node = $dom->createElement('aluno');
		$child_node_id = $dom->createElement('id', $row["id"]);
		$aluno_node->appendChild($child_node_id);
		$child_node_nome = $dom->createElement('nome', $row["nome"]);
		$aluno_node->appendChild($child_node_nome);
		$root->appendChild($aluno_node);
	}

	$dom->appendChild($root);
	$dom->save($xml_file_name);

	$data = file_get_contents($xml_file_name) or die("Failed to load");
	print_r($data);
```

E o Webservice de obtenção dos dados dos alunos “getAlunos.php”:

getAlunos.php

REST > db > getAlunos.php

```php
<?php
    function getAlunos(){

        try {
            include_once("config/conn.php");

            $PDO->beginTransaction();

            $query = "SELECT * FROM alunos";

            $result = $PDO->query($query)->fetchAll();

            return $result;
        }catch(PDOException $e){
            $PDO->rollBack();
            var_dump($e);
        }
    }

?>
```

Desenvolveu-se um controlador REST implementado no ficheiro “**RestController.php**”:

RestController.php

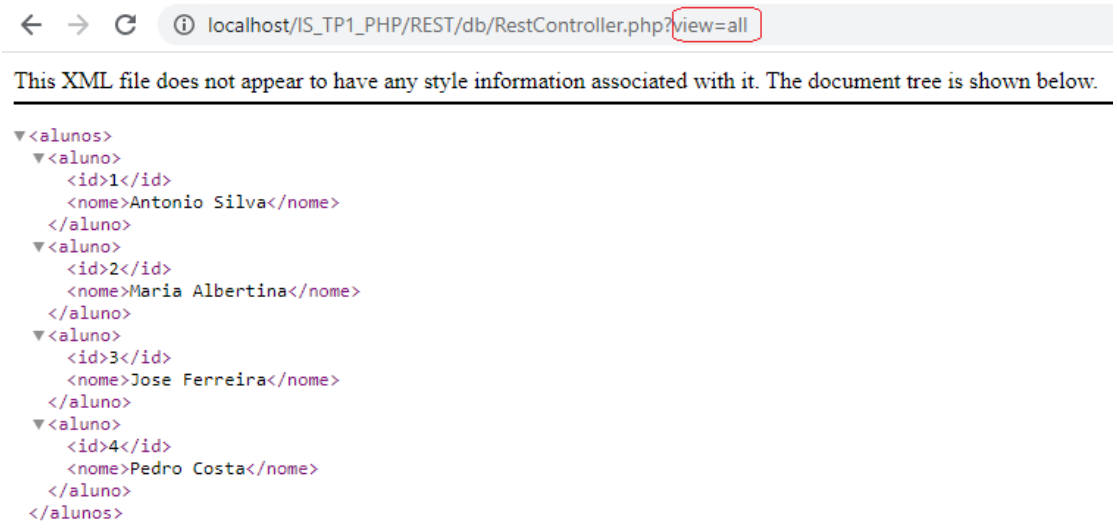
```php
<?php
require_once("AlunosRestHandler.php");

$view = "";
if(isset($_GET["view"]))
    $view = $_GET["view"];

switch($view){

    case "all":
        // to handle REST Url /mobile/list/
        $AlunosRestHandler = new AlunosRestHandler();
        $AlunosRestHandler->getAllAlunos();
        break;
}

?>
```

No ficheiro **"SimpleRest.php"** apresenta-se a implementação de um pedido simples REST:

SimpleRest.php

```php
<?php

class SimpleRest {

    private $httpVersion = "HTTP/1.1";

    public function setHttpHeaders($contentType, $statusCode){

        $statusMessage = $this -> getHttpStatusMessage($statusCode);

        header($this->httpVersion. " ". $statusCode ." ". $statusMessage);
        header("Content-Type:". $contentType);
    }

    public function getHttpStatusMessage($statusCode){
        $httpStatus = array(
            100 => 'Continue',
            101 => 'Switching Protocols',
            200 => 'OK',
            201 => 'Created',
            202 => 'Accepted',
            203 => 'Non-Authoritative Information',
            204 => 'No Content',
            205 => 'Reset Content',
            206 => 'Partial Content',
            300 => 'Multiple Choices',
            301 => 'Moved Permanently',
            302 => 'Found',
            303 => 'See Other',
            304 => 'Not Modified',
            305 => 'Use Proxy',
            306 => '(Unused)',
            307 => 'Temporary Redirect',
            400 => 'Bad Request',
            401 => 'Unauthorized',
            402 => 'Payment Required',
            403 => 'Forbidden',
            404 => 'Not Found',
            405 => 'Method Not Allowed',
            406 => 'Not Acceptable',
            407 => 'Proxy Authentication Required',
            408 => 'Request Timeout',
            409 => 'Conflict',
            410 => 'Gone',
            411 => 'Length Required',
            412 => 'Precondition Failed',
            413 => 'Request Entity Too Large',
            414 => 'Request-URI Too Long',
            415 => 'Unsupported Media Type',
            416 => 'Requested Range Not Satisfiable',
            417 => 'Expectation Failed',
            500 => 'Internal Server Error',
            501 => 'Not Implemented',
            502 => 'Bad Gateway',
            503 => 'Service Unavailable',
            504 => 'Gateway Timeout',
            505 => 'HTTP Version Not Supported');
        return ($httpStatus[$statusCode]) ? $httpStatus[$statusCode] : $status[500];
    }
}
```

Teste do Webservice (**o endpoint terá que ter a view=all**)



2.3.2 Importação do ficheiro books.xml via REST usando o parser DOM e/ou SAX

Considere o ficheiro "**BooksRestHandler.php**":

BooksRestHandler.php

REST > DOM > BooksRestHandler.php

```php
<?php
require_once("SimpleRest.php");
require_once("readFile.php");

class BooksRestHandler extends SimpleRest {

	function getAllBooks() {

		$rawData = getDataFile();
			if(empty($rawData)) {
				$statusCode = 404;
				//$rawData = array('error' => 'No books found!');
			} else {
				$statusCode = 200;
			}

			$requestContentType = $_SERVER['HTTP_ACCEPT'];
			$this ->setHttpHeaders($requestContentType, $statusCode);

			if(strpos($requestContentType,'text/html') !== false){
		//print_r($rawData);
				header('Content-Type: text/html');
				$response = $this->encodeXml($rawData);
				return $response;
			}
		}

		public function encodeXml($responseData) {

			print_r($responseData);
		}
	}
?>
```

Eo ficheiro "**readFile.php" para leitura do ficheiro via DOM:**

readFile.php

```php
<?php

$dom = new domDocument;

    $dom->load("files/books.xml");

    $dom->preserveWhiteSpace = false;

    $tables = $dom->getElementsByTagName('catalog');

    $rows = $tables->item(0)->getElementsByTagName('book');

    foreach ($rows as $row) {

        $author = $row->getElementsByTagName('author');
        $title = $row->getElementsByTagName('title');
        $genre = $row->getElementsByTagName('genre');
        $price = $row->getElementsByTagName('price');
        $publish_date = $row->getElementsByTagName('publish_date');
        $description = $row->getElementsByTagName('description');


        echo 'author: '.$author->item(0)->nodeValue.'<br />';
        echo 'title: '.$title->item(0)->nodeValue.'<br />';
        echo 'genre: '.$genre->item(0)->nodeValue;
        echo 'price: '.$price->item(0)->nodeValue.'<br />';
        echo 'publish_date: '.$publish_date->item(0)->nodeValue.'<br />';
        echo 'description: '.$description->item(0)->nodeValue;
        echo '<hr />';
    }

?>
```

O ficheiro “**RestController.php” e o “SimpleRest.php”:**

RestController.php

```php
<?php

require_once("BooksRestHandler.php");

$view = "";
if(isset($_GET["view"]))
    $view = $_GET["view"];

switch($view){

    case "all":
        // to handle REST Url /mobile/list/
        $BooksRestHandler = new BooksRestHandler();
        $BooksRestHandler->getAllBooks();
        break;
}

?>
```

SimpleRest.php

```php
<?php

class SimpleRest {

    private $httpVersion = "HTTP/1.1";

    public function setHttpHeaders($contentType, $statusCode){

        $statusMessage = $this -> getHttpStatusMessage($statusCode);

        header($this->httpVersion. " ". $statusCode ." ". $statusMessage);
        header("Content-Type:". $contentType);
    }

    public function getHttpStatusMessage($statusCode){
        $httpStatus = array(
            100 => 'Continue',
            101 => 'Switching Protocols',
            200 => 'OK',
            201 => 'Created',
            202 => 'Accepted',
            203 => 'Non-Authoritative Information',
            204 => 'No Content',
            205 => 'Reset Content',
            206 => 'Partial Content',
            300 => 'Multiple Choices',
            301 => 'Moved Permanently',
            302 => 'Found',
            303 => 'See Other',
            304 => 'Not Modified',
            305 => 'Use Proxy',
            306 => '(Unused)',
            307 => 'Temporary Redirect',
            400 => 'Bad Request',
            401 => 'Unauthorized',
            402 => 'Payment Required',
            403 => 'Forbidden',
            404 => 'Not Found',
            405 => 'Method Not Allowed',
            406 => 'Not Acceptable',
            407 => 'Proxy Authentication Required',
            408 => 'Request Timeout',
            409 => 'Conflict',
            410 => 'Gone',
            411 => 'Length Required',
            412 => 'Precondition Failed',
            413 => 'Request Entity Too Large',
            414 => 'Request-URI Too Long',
            415 => 'Unsupported Media Type',
            416 => 'Requested Range Not Satisfiable',
            417 => 'Expectation Failed',
            500 => 'Internal Server Error',
            501 => 'Not Implemented',
            502 => 'Bad Gateway',
            503 => 'Service Unavailable',
            504 => 'Gateway Timeout',
            505 => 'HTTP Version Not Supported');
        return ($httpStatus[$statusCode]) ? $httpStatus[$statusCode] : $status[500];
    }
}
```

Teste do Webservice:

localhost/IS_TP1_PHP/REST/DOM/RestController.php

author: Gambardella, Matthew
title: XML Developer's Guide
genre: Computerprice: 44.95
publish_date: 2000-10-01
description: An in-depth look at creating applications with XML.

---

author: Ralls, Kim
title: Midnight Rain
genre: Fantasyprice: 5.95
publish_date: 2000-12-16
description: A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.

---

author: Corets, Eva
title: Maeve Ascendant
genre: Fantasyprice: 5.95
publish_date: 2000-11-17
description: After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.

---

author: Corets, Eva
title: Oberon's Legacy
genre: Fantasyprice: 5.95
publish_date: 2001-03-10
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.

---

author: Corets, Eva
title: The Sundered Grail
genre: Fantasyprice: 5.95
publish_date: 2001-09-10
description: The two daughters of Maeve, half-sisters, battle one another for control of England. Sequel to Oberon's Legacy.

---

author: Randall, Cynthia
title: Lover Birds
genre: Romanceprice: 4.95
publish_date: 2000-09-02
description: When Carla meets Paul at an ornithology conference, tempers fly as feathers get ruffled.

---

author: Thurman, Paula
title: Splish Splash
genre: Romanceprice: 4.95
publish_date: 2000-11-02
description: A deep sea diver finds true love twenty thousand leagues beneath the sea.

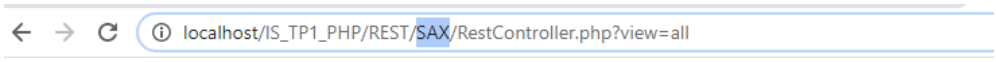
Considere o ficheiro "**readFile.php**":

readFile.php

REST > SAX > readFile.php

```php
<?php

    $tutors   = array();
    $elements   = null;

    function startElements($parser, $name, $attrs) {
        global $tutors, $elements;

        if(!empty($name)) {
            if ($name == 'BOOK') {
                $tutors []= array();
            }
            $elements = $name;
        }
    }

    function endElements($parser, $name) {
        global $elements;

        if(!empty($name)) {
            $elements = null;
        }
    }

    function characterData($parser, $data) {
        global $tutors, $elements;

        if(!empty($data)) {
            if ($elements == 'AUTHOR' || $elements == 'TITLE' ||  $elements == 'GENRE' ||  $elements == 'PRICE' || $elements == 'PUBLISH_DATE' || $elements == 'DESCRIPTION') {
                $tutors[count($tutors)-1][$elements] = trim($data);
            }
        }
    }

    function getDataFile() {
        global $tutors, $elements;

    $parser = xml_parser_create();

    xml_set_element_handler($parser, "startElements", "endElements");
    xml_set_character_data_handler($parser, "characterData");

    if (!($handle = fopen('files/books.xml', "r"))) {
        die("could not open XML input");
    }

    while($data = fread($handle, 8096)) {
        xml_parse($parser, $data);
    }

    xml_parser_free($parser);

    $dataValue = "";
    foreach($tutors as $course) {
        $dataValue .= "author - ".$course['AUTHOR'].'<br/>';
        $dataValue .= "title - ".$course['TITLE'].'<br/>';
        $dataValue .= "genre - ".$course['GENRE'].'<br/>';
        $dataValue .= "price - ".$course['PRICE'].'<br/>';
        $dataValue .= "publish_date - ".$course['PUBLISH_DATE'].'<br/>';
        $dataValue .= "description - ".$course['DESCRIPTION'].'<hr/>';
    }

    return $dataValue;

    }
?>
```

Execução do webservice usando SAX (o e**ndpoint terá que ter "?view=all"):**



author - Gambardella, Matthew
title - XML Developer's Guide
genre - Computer
price - 44.95
publish_date - 2000-10-01
description - with XML.

---

author - Ralls, Kim
title - Midnight Rain
genre - Fantasy
price - 5.95
publish_date - 2000-12-16
description - of the world.

---

author - Corets, Eva
title - Maeve Ascendant
genre - Fantasy
price - 5.95
publish_date - 2000-11-17
description - foundation for a new society.

---

author - Corets, Eva
title - Oberon's Legacy
genre - Fantasy
price - 5.95
publish_date - 2001-03-10
description - Ascendant.

---

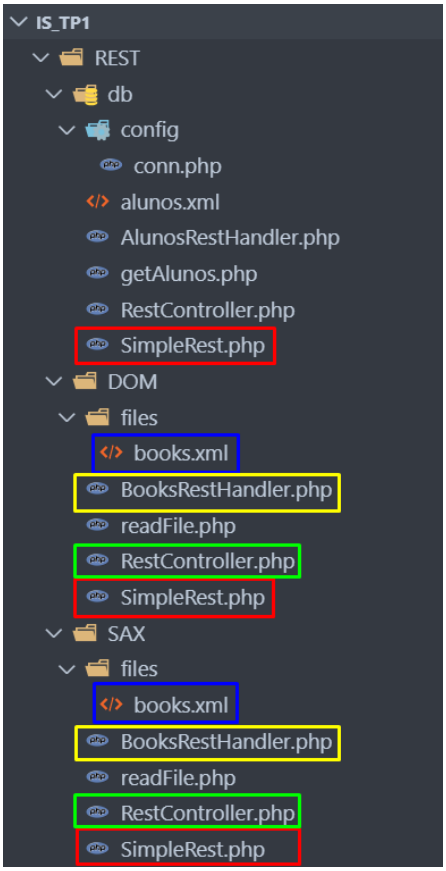
author - Corets, Eva
title - The Sundered Grail
genre - Fantasy
price - 5.95
publish_date - 2001-09-10
description - Oberon's Legacy.

---

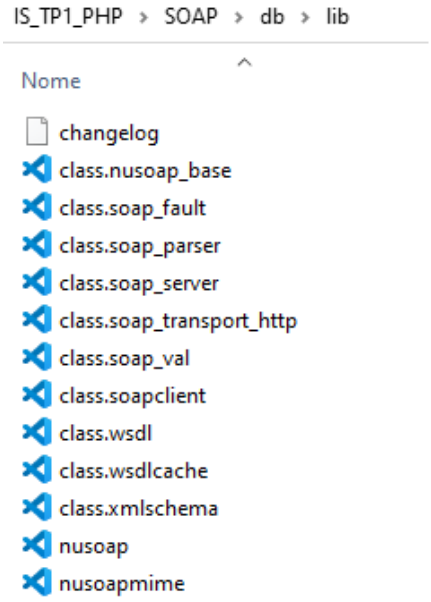
author - Randall, Cynthia
title - Lover Birds
genre - Romance
price - 4.95
publish_date - 2000-09-02
description - conference, tempers fly as feathers get ruffled.

2.3.3 Execução de webservice via SOAP para leitura de dados de uma base de dados

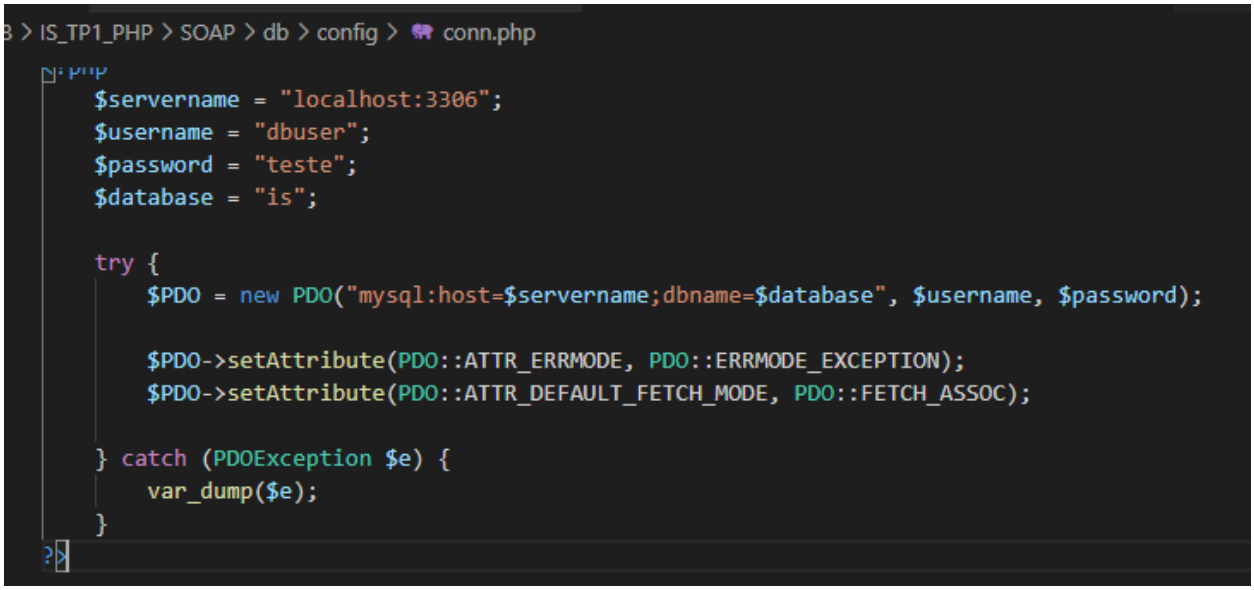
Considere a seguinte estrutura de ficheiros na diretoria SOAP, os ficheiros com a mesma cor são iguais:



Considere a inclusão do package “nusoap.php”:



Considere as configurações de acesso à base de dados:



Considere o ficheiro "**client.php**":

E o ficheiro "**getAlunos.php**":

getAlunos.php

SOAP > db > getAlunos.php

```php
<?php

    function getAlunos(){

        try {

            include_once("config/conn.php");

            $PDO->beginTransaction();

            $query = "SELECT * FROM alunos";

            $result = $PDO->query($query)->fetchAll();

            return $result;
        }catch(PDOException $e){
            $PDO->rollBack();
            var_dump($e);
        }
    }
?>
```

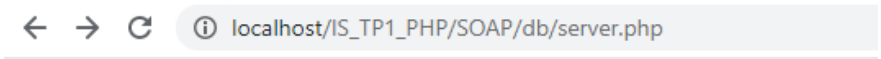
E o ficheiro “**server.php**”:

server.php

SOAP > db > server.php

```php
<?php
    require_once("lib/nusoap.php");
    include_once("getAlunos.php");

    $server = new nusoap_server();

    $server->configureWSDL('urn:Server');
    $server->wsdl->schemaTargetNamespace = 'urn:Server';


    function getData($file){
        return getDataFromFile();
    }

    function getDataFromFile(){
        $data = getAlunos();

                $dom = new DOMDocument();
                $dom->encoding = 'utf-8';
                $dom->xmlVersion = '1.0';
                $dom->formatOutput = true;
                $xml_file_name = 'alunos.xml';
                $root = $dom->createElement('alunos');

                foreach ($data as $row) {
                    $aluno_node = $dom->createElement('aluno');
                    $child_node_id = $dom->createElement('id', $row["id"]);
                    $aluno_node->appendChild($child_node_id);
                    $child_node_nome = $dom->createElement('nome', $row["nome"]);
                    $aluno_node->appendChild($child_node_nome);
                    $root->appendChild($aluno_node);
                }

                $dom->appendChild($root);
                $dom->save($xml_file_name);

                $dataDom = file_get_contents($xml_file_name) or die("Failed to load");

                return $dataDom;
    }

    $server->register (
        'getData',
        array("file"=>"xsd:string"),
        array("return"=>"xsd:string"),
        'urn:Server.getData',
        'urn:Server.getData',
        'rpc',
        'encoded',
        'buscar dados a uma base de dados'
    );

    $HTTP_RAW_POST_DATA = isset( $HTTP_RAW_POST_DATA ) ? $HTTP_RAW_POST_DATA : file_get_contents("php://input");
        $server->service($HTTP_RAW_POST_DATA);
?>
```

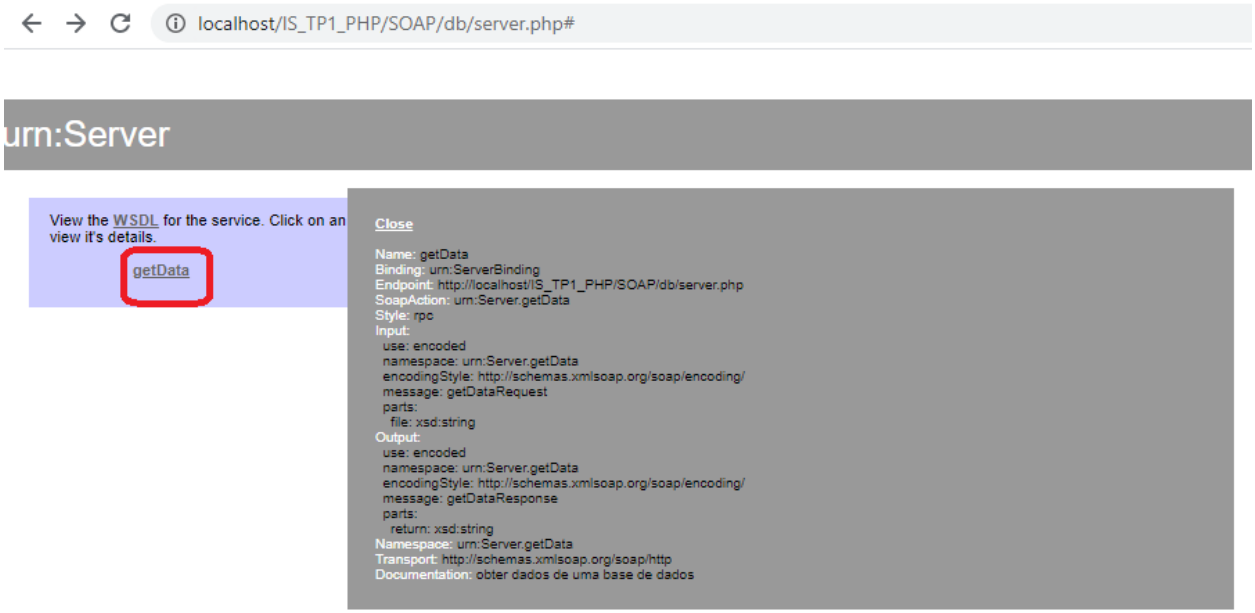
Invocação do webservice:



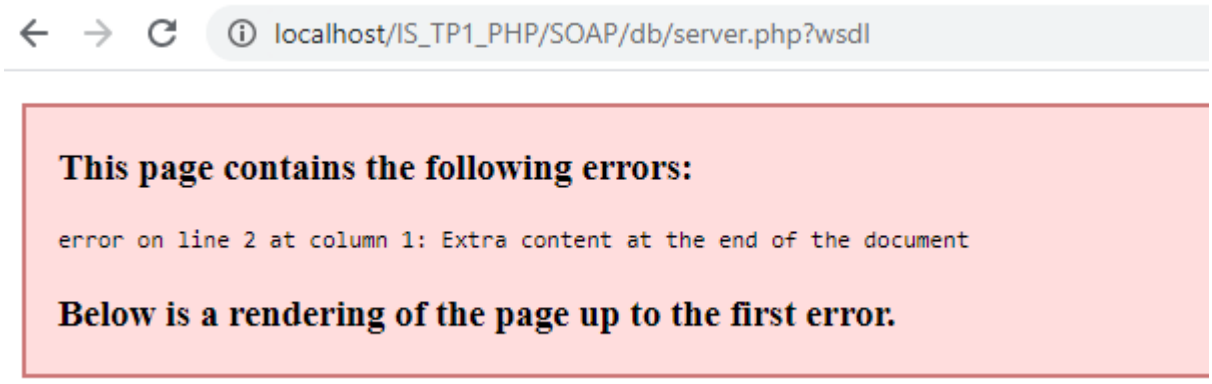
urn:Server

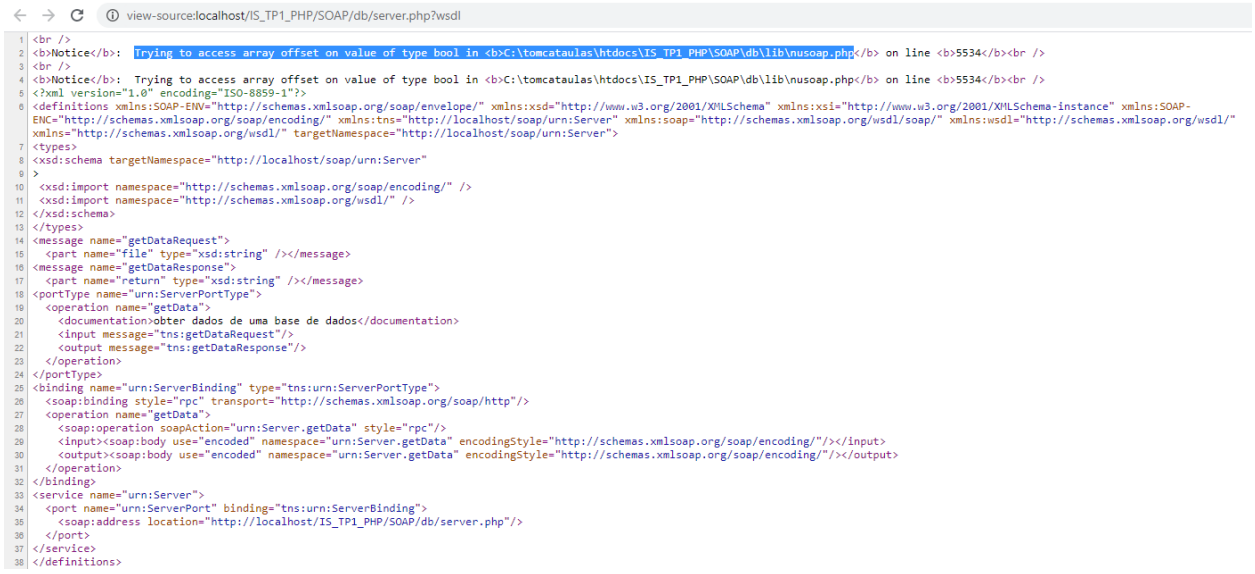
View the WSDL for the service. Click on an operation name to view it's details.

getData



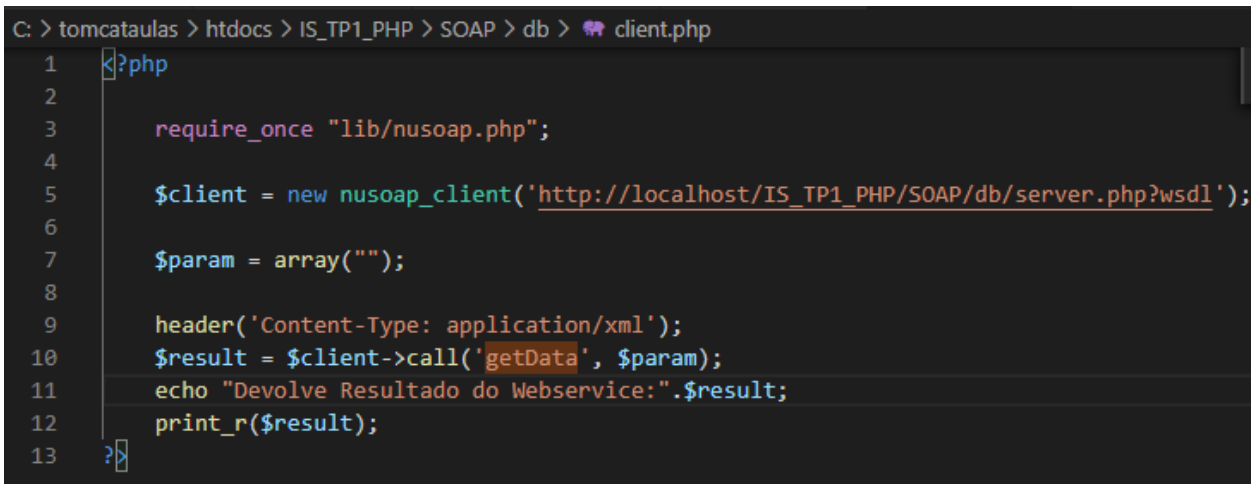
Se aparecer este erro e analisarem o código fonte:

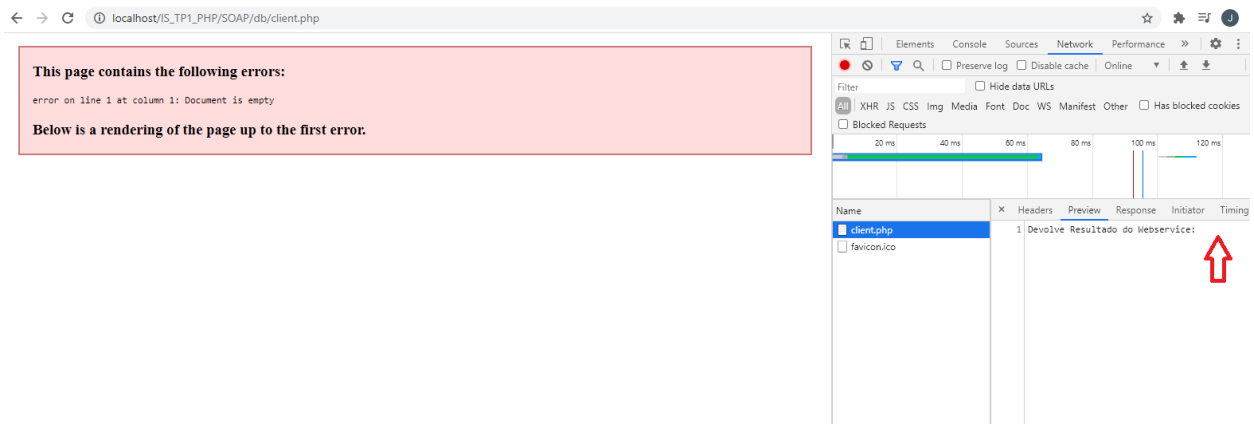




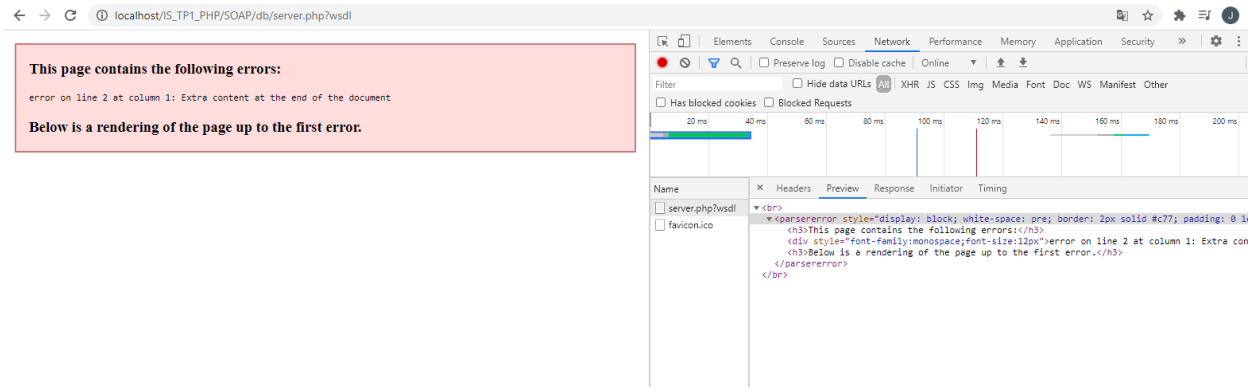
Ao analisar a linha 5534, verifica-se que o problema está na construção dos elementos XML:

Acontece que o webservice devolve um valor vazio/nulo em vez de um documento XML. Para testar esta situação acrescentou-se uma linha de código para devolver o resultado:





Ao verificar a execução do server.php:



O problema encontra-se na devolução de resultados da base de dados através do comando fetcall()

```
htdocs > IS_TP1_PHP > SOAP > db > getAlunos.php
<?php

function getAlunos(){

    try {
        require_once("config/conn.php");

        $PDO->beginTransaction();

        $query = "SELECT * FROM alunos";

        $result = $PDO->query($query)->fetchAll();
        return $result;
    }catch(PDOException $e){
        $PDO->rollBack();
        var_dump($e);
    }
}
```

Para resolver o problema, deverá verificar os packages de instalação do pacote PDO no PHP, ou então proceder a uma forma diferente de acesso aos dados da base de dados e à devolução de resultados de uma query.

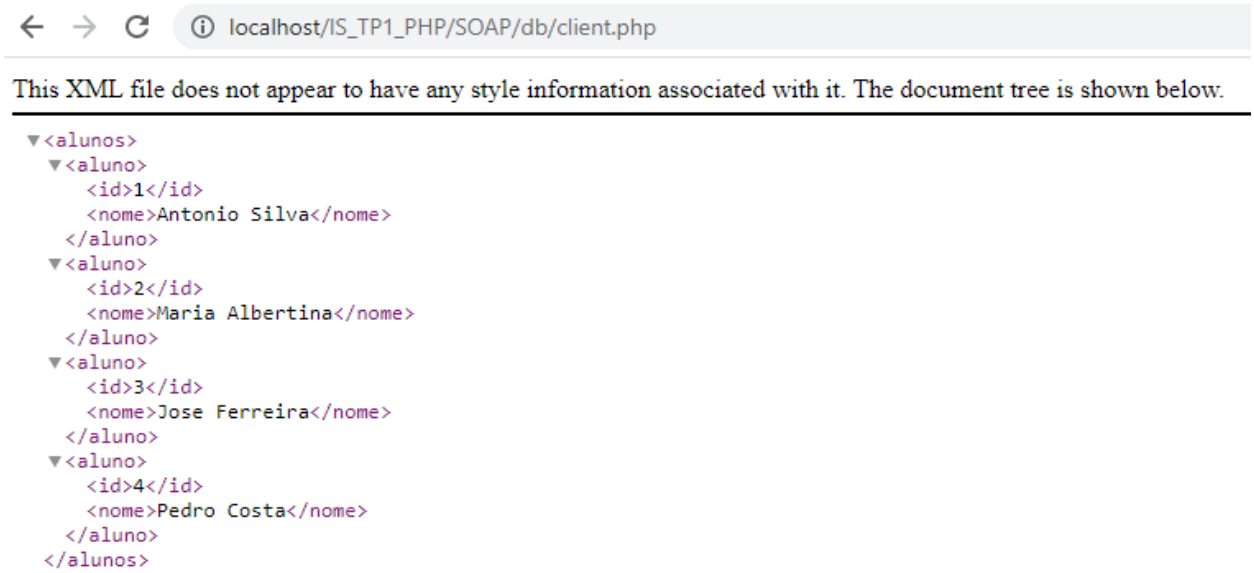
Uma solução poderá passar pela adição da extensão ou retirar o comentário da linha para o pdo_mysql.

php - Bloco de notas

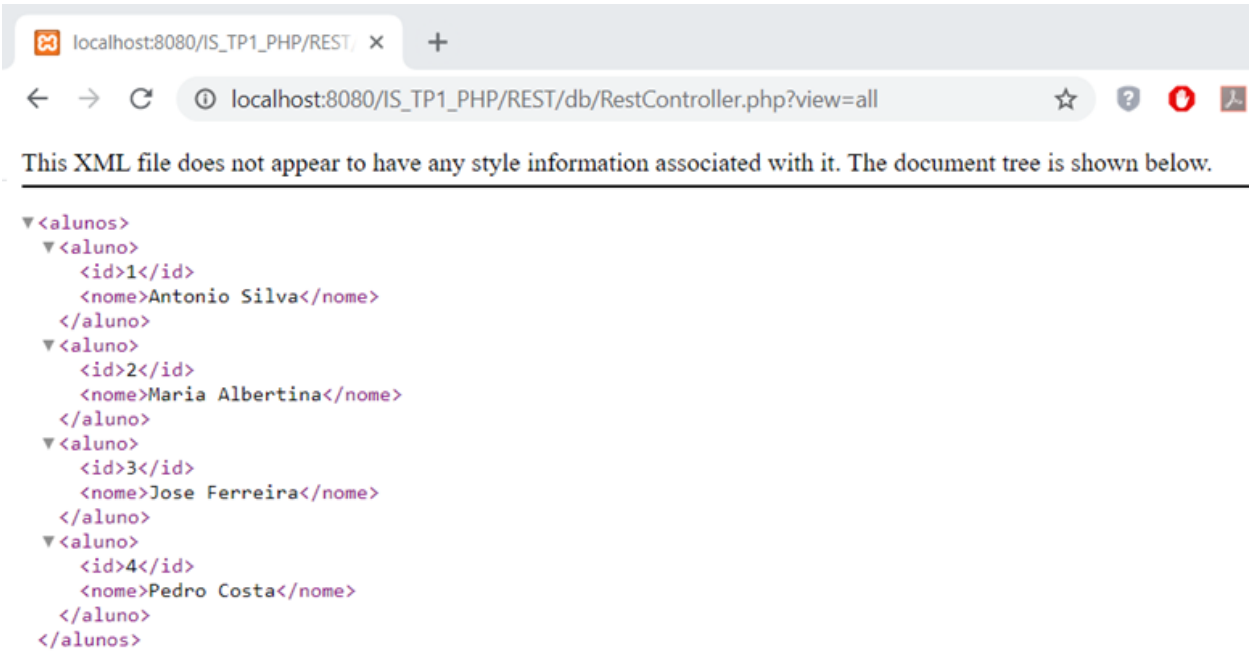
Ficheiro Editar Formatar Ver Ajuda

```
; move to the new ('extension=<ext>) syntax.
;
; Notes for Windows environments :
;
; - Many DLL files are located in the extensions/ (PHP 4) or ext/ (PHP 5+)
;   extension folders as well as the separate PECL DLL download (PHP 5+).
;   Be sure to appropriately set the extension_dir directive.
;
extension=bz2
extension=curl
;extension=ffi
;extension=ftp
extension=fileinfo
extension=gd2
extension=gettext
;extension=gmp
;extension=intl
;extension=imap
;extension=ldap
extension=mbstring
extension=exif      ; Must be after mbstring as it depends on it
extension=mysqli
;extension=oci8_12c  ; Use with Oracle Database 12c Instant Client
;extension=odbc
;extension=openssl
;extension=pdo_firebird
extension=pdo_mysql
;extension=pdo_oci
;extension=pdo_odbc
;extension=pdo_pgsql
extension=pdo_sqlite
;extension=pgsql
;extension=shmop

; The MIBS data available in the PHP distribution must be installed.
; See http://www.php.net/manual/en/snmp.installation.php
;extension=snmp

;extension=soap
;extension=sockets
:extension=sodium
```

O resultado da invocação do webservice é a seguinte:



E o do acesso à BD:



2.3.4 Implementação de webservice via SOAP para leitura de ficheiro books.xml usando o parser DOM

Considere o ficheiro “cliente.php”:

htdocs > IS_TP1_PHP > SOAP > DOM > client.php

```php
<?php
    require_once("lib/nusoap.php");

    $client = new nusoap_client('http://localhost/IS_TP1_PHP/SOAP/DOM/server.php?wsdl');

    $param = array('file'=>'books.xml');

    $result = $client->call('getData', $param);

    print_r($result);
?>
```

E o ficheiro “**readFile.php”**

readFile.php

```php
<?php

    function getDataFile() {
    $dom = new domDocument;

    $dom->load("files/books.xml");

    $dom->preserveWhiteSpace = false;

    $tables = $dom->getElementsByTagName('catalog');

    $rows = $tables->item(0)->getElementsByTagName('book');

    $data = "";

    foreach ($rows as $row) {
        $author = $row->getElementsByTagName('author');
        $title = $row->getElementsByTagName('title');
        $genre = $row->getElementsByTagName('genre');
        $price = $row->getElementsByTagName('price');
        $publish_date = $row->getElementsByTagName('publish_date');
        $description = $row->getElementsByTagName('description');
        $data .= 'author: '.$author->item(0)->nodeValue.'<br />';
        $data .= 'title: '.$title->item(0)->nodeValue.'<br />';
        $data .= 'genre: '.$genre->item(0)->nodeValue;
        $data .= 'price: '.$price->item(0)->nodeValue.'<br />';
        $data .= 'publish_date: '.$publish_date->item(0)->nodeValue.'<br />';
        $data .= 'description: '.$description->item(0)->nodeValue;
        $data .= '<hr />';
    }

    return $data;

    }

?>
```

O resultado da invocação do webservice é a seguinte:

localhost/IS_TP1_PHP/SOAP/DOM/client.php

author: Gambardella, Matthew
title: XML Developer's Guide
genre: Computerprice: 44.95
publish_date: 2000-10-01
description: An in-depth look at creating applications with XML.

---

author: Ralls, Kim
title: Midnight Rain
genre: Fantasyprice: 5.95
publish_date: 2000-12-16
description: A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.

---

author: Corets, Eva
title: Maeve Ascendant
genre: Fantasyprice: 5.95
publish_date: 2000-11-17
description: After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.

---

author: Corets, Eva
title: Oberon's Legacy
genre: Fantasyprice: 5.95
publish_date: 2001-03-10
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.

---

author: Corets, Eva
title: The Sundered Grail
genre: Fantasyprice: 5.95
publish_date: 2001-09-10
description: The two daughters of Maeve, half-sisters, battle one another for control of England. Sequel to Oberon's Legacy.

---

author: Randall, Cynthia
title: Lover Birds
genre: Romanceprice: 4.95
publish_date: 2000-09-02
description: When Carla meets Paul at an ornithology conference, tempers fly as feathers get ruffled.

---

author: Thurman, Paula
title: Splish Splash
genre: Romanceprice: 4.95
publish_date: 2000-11-02
description: A deep sea diver finds true love twenty thousand leagues beneath the sea.

---

author: Knorr, Stefan

2.3.5 Implementação de webservice via SOAP para leitura de ficheiro books.xml usando o parser SAX

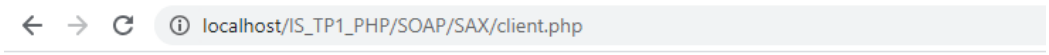
Considere o ficheiro “client.php”:

```
tdocs > IS_TP1_PHP > SOAP > SAX > client.php
<?php
    require_once("lib/nusoap.php");

    $client = new nusoap_client('http://localhost/IS_TP1_PHP/SOAP/SAX/server.php?wsdl');

    $param = array('file'=>'books.xml');

    $result = $client->call('getData', $param);

    print_r($result);
?>
```

O Ficheiro “**readFile.php**”

```
readFile.php
SOAP > SAX > readFile.php
<?php

    $tutors   = array();
    $elements   = null;

    function startElements($parser, $name, $attrs) {
        global $tutors, $elements;

        if(!empty($name)) {
            if ($name == 'BOOK') {
                $tutors []= array();
            }
            $elements = $name;
        }
    }

    function endElements($parser, $name) {
        global $elements;

        if(!empty($name)) {
            $elements = null;
        }
    }

    function characterData($parser, $data) {
        global $tutors, $elements;

        if(!empty($data)) {
            if ($elements == 'AUTHOR' || $elements == 'TITLE' ||  $elements == 'GENRE' ||  $elements == 'PRICE' || $elements == 'PUBLISH_DATE' || $elements == 'DESCRIPTION') {
                $tutors[count($tutors)-1][$elements] = trim($data);
            }
        }
    }

    function getDataFile() {
        global $tutors, $elements;

    $parser = xml_parser_create();

    xml_set_element_handler($parser, "startElements", "endElements");
    xml_set_character_data_handler($parser, "characterData");

    if (!($handle = fopen('files/books.xml', "r"))) {
        die("could not open XML input");
    }

    while($data = fread($handle, 8096)) {
        xml_parse($parser, $data);
    }

    xml_parser_free($parser);

    $dataValue = "";
    foreach($tutors as $course) {
        $dataValue .= "author - ".$course['AUTHOR'].'<br/>';
        $dataValue .= "title - ".$course['TITLE'].'<br/>';
        $dataValue .= "genre - ".$course['GENRE'].'<br/>';
        $dataValue .= "price - ".$course['PRICE'].'<br/>';
        $dataValue .= "publish_date - ".$course['PUBLISH_DATE'].'<br/>';
        $dataValue .= "description - ".$course['DESCRIPTION'].'<hr/>';
    }

    return $dataValue;

    }
?>
```

O resultado da invocação do webservice é a seguinte:



author - Gambardella, Matthew
title - XML Developer's Guide
genre - Computer
price - 44.95
publish_date - 2000-10-01
description - with XML.

---

author - Ralls, Kim
title - Midnight Rain
genre - Fantasy
price - 5.95
publish_date - 2000-12-16
description - of the world.

---

author - Corets, Eva
title - Maeve Ascendant
genre - Fantasy
price - 5.95
publish_date - 2000-11-17
description - foundation for a new society.

---

author - Corets, Eva
title - Oberon's Legacy
genre - Fantasy
price - 5.95
publish_date - 2001-03-10
description - Ascendant.

---

author - Corets, Eva
title - The Sundered Grail
genre - Fantasy
price - 5.95
publish_date - 2001-09-10