XML Processing

Exporting and Importing Data from XML format

SoftUni Team

Technical Trainers

Software University

SoftUni Foundation

Software University

http://softuni.bg

Table of Content

Questions

SoftUni Foundation

sli.do

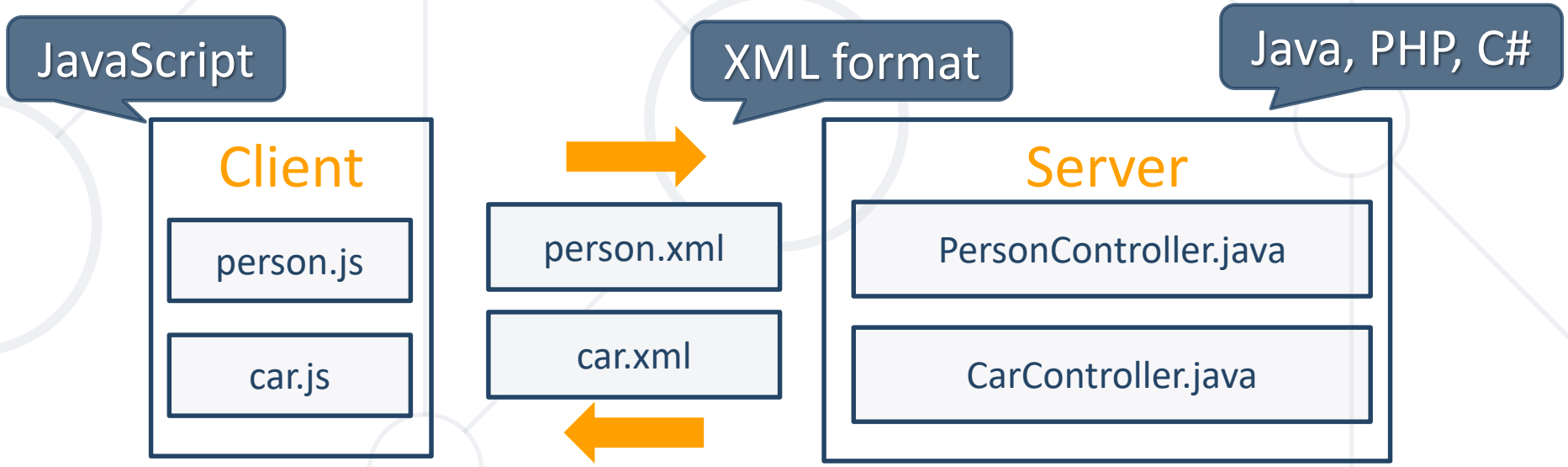
#JavaDb

XML Processing

Exporting and Importing Data From XML Format

XML Specifics

- EXtensible Mark-up Language
  - Lightweight format that is used for **data interchanging**
  - XML is language independent
- Primarily used to transmit data between a server and web application



XML Markup and Content

SoftUni Foundation

- An XML document consists of strings that:
  - Constitute **markup** – usually begin with **<** and end with **>**
  - Are **content** – placed between markup(**tags**)
    - **E.g.:**

person.xml

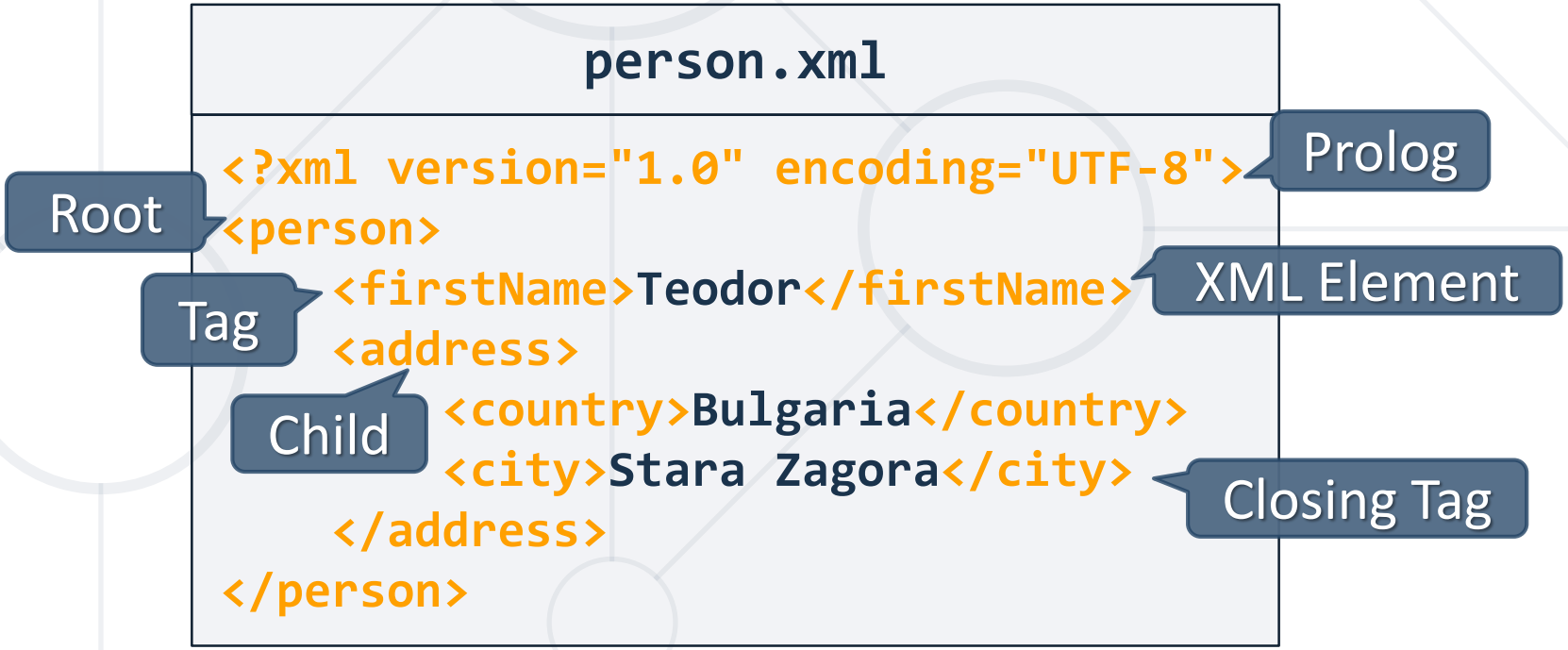
```
<?xml version="1.0" encoding="UTF-8">
<person>
    <firstName>Teodor</firstName>
</person>
```

Markup tags for Person Object

Content (Person Name)

XML Structure

- XML documents are formed as **element trees**
- An XML tree starts at a **root element** and branches from the root to **sub elements**
  - All elements can have child ones:



XML Structure (2)

SoftUni Foundation

person.xml

```
<?xml version="1.0" encoding="UTF-8">
<person>
    <phoneNumbers>
        <phoneNumber>
            <number>08983248798</number>
        </phoneNumber>
        <phoneNumber>
            <number>08983243143</number>
        </phoneNumber>
    </phoneNumbers>
</person>
```

Wrapper

JAXB

Parsing XML to Java Objects

JAXB

- Processes the schema of the XML document into a set of Java classes that represent it
- Generates compact and readable XML output

pom.xml

```
<dependency>
    <groupId>com.sun.xml.bind</groupId>
    <artifactId>jaxb-impl</artifactId>
</dependency>
```

JAXB Basics

- **Marshalling** - converting a Java Object to XML
- **Unmarshalling** - converting XML to Java Object
- We need to annotate the Java Object to provide instructions for XML creation:

AddressDto.java

```
@XmlRootElement(name = "address")
@XmlAccessorType(XmlAccessType.FIELD)
public class AddressDto implements Serializable
{
    @XmlAttribute(name = "country")
    private String country;

    @XmlElement(name = "city")
    private String city;
}
```

JAXB Annotations

SoftUni Foundation

- @XmlRootElement – defines XML root object
- @XmlAccessorType
  - XmlAccessType.**FIELD**, XmlAccessType.**PROPERTY**, **XmlAccessType.PUBLIC_MEMBER**
- @XmlAttribute – marks the field as an attribute to the object
- @XmlElement – marks the field as an element
- @XmlElementWrapper(name = "…") – wraps the array of objects
- @XmlTransient – the field won't be exported/imported

JAXB Initialization

- **JAXBContext** objects are responsible for the XML manipulations
- JAXBContext.newInstance(object.getClass()) - creates an **instance** of JAXBContext
- **object.getClass** is the class that we will export/import
  - E.g. User, Address, Employee…

XMLParser.java

```
this.jaxbContext = JAXBContext.newInstance(object.getClass());
```

Export Single Object to XML - Example

SoftUni Foundation

User.java

```
@XmlRootElement
@XmlAccessorType(XmlAccessType.FIELD)
public class User {
    @XmlElement(name = "name")
    private String name;
    @XmlElement(name = "age")
    private Integer age;

    public String getName() {
        return name;
    }
// Constructor, getters, setters
```

XMLParser.java

```
JAXBContext context = JAXBContext.newInstance(User.class);
Marshaller marshaller = context.createMarshaller();
marshaller.marshal(user, new File("users.xml"));
```

Creates XML file "users.xml"

users.xml

```
<?xml version="1.0" encod
ing="UTF-8" standalone="y
es"?>
<user>
    <name>New User</name>
    <age>18</age>
</user>
```

Export Single Object to XML – Example 2

SoftUni Foundation

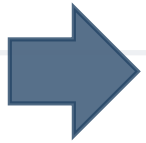
AddressDto.java

```
@XmlRootElement(name = "address")
@XmlAccessorType(XmlAccessType.FIELD)
public class AddressDto implements Serializable {

    @XmlAttribute(name = "country")
    private String country;

    @XmlElement(name = "city")
    private String city;
}
```

Object attribute



address.xml

```
<?xml version="1.0"
encoding="UTF-8"?>
<address country="Bulgaria">
    <city>Sofia</city>
</address>
```

XMLParser.java

```
Marshaller jaxbMarshaller = jaxbContext.createMarshaller();
jaxbMarshaller.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
OutputStream outputStream = new FileOutputStream(fileName);
BufferedWriter bfw =
    new BufferedWriter(new OutputStreamWriter(outputStream));
jaxbMarshaller.marshal(object, bfw);
```

Format XML output (Analogically to setPrettyPrinting in JSON parsing)

Export Single Object to XML

SoftUni Foundation

AddressDto.java

```
@XmlRootElement(name = "address")
@XmlAccessorType(XmlAccessType.FIELD)
public class AddressJsonDto implements Serializable {

    @XmlAttribute(name = "country")
    private String country;

    @XmlElement(name = "city")
    private String city;
}
```

address.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<address country="Bulgaria">
    <city>Sofia</city>
</address>
```

Export Multiple Objects to XML

SoftUni Foundation

AddressesDto.java

```
@XmlRootElement(name = "addresses")
@XmlAccessorType(XmlAccessType.FIELD)
public class AddressesDto {

    @XmlElement(name = "address")
    private List<AddressDto> addressJsonDtos;
}
```

XMLParser.java

```
AddressesDto addressDtos = new AddressesDto();
jaxbMarshaller.marshal(addressesDto, bfw);
```

Export Multiple Objects to XML (2)

SoftUni Foundation

XMLParser.java

```
AddressesDto addressDtos = new AddressesDto();
jaxbMarshaller.marshal(addressesDto, bfw);
```

addresses.json

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<addresses>
    <address country="Bulgaria">
        <city>Sofia</city>
    </address>
    <address country="Spain">
        <city>Barcelona</city>
    </address>
</addresses>
```

Import Single Object from XML

SoftUni Foundation

AddressDto.java

```
@XmlRootElement(name = "address")
@XmlAccessorType(XmlAccessType.FIELD)
public class AddressDto implements Serializable {

    @XmlAttribute(name = "country")
    private String country;

    @XmlElement(name = "city")
    private String city;
}
```

XMLParser.java

```
JAXBContext jaxbContext = JAXBContext.newInstance(AddressDto.class);
InputStream inputStream = getClass().getResourceAsStream("/files/input/xml/
address.xml");
BufferedReader bfr = new BufferedReader(new InputStreamReader(inputStream));
Unmarshaller unmarshaller = jaxbContext.createUnmarshaller();
AddressDto addressDto = (AddressDto) unmarshaller.unmarshal(bfr);
```

Creates Object

Import Single Object from XML

SoftUni Foundation

AddressDto.java

```
@XmlRootElement(name = "address")
@XmlAccessorType(XmlAccessType.FIELD)
public class AddressDto implements
Serializable {

    @XmlAttribute(name = "country")
    private String country;

    @XmlElement(name = "city")
    private String city;
}
```

address.xml

```
<?xml version="1.0" encoding=
"UTF-8" standalone="yes"?>
<address country="Bulgaria">
    <city>Sofia</city>
</address>
```

Import Multiple Objects to XML

SoftUni Foundation

XMLParser.java

```
JAXBContext jaxbContext = JAXBContext.newInstance(AddressesDto.class);
InputStream inputStream = getClass().getResourceAsStream("/files/input/xml/addresses.xml");
BufferedReader bfr = new BufferedReader(new InputStreamReader(inputStream));
Unmarshaller unmarshaller = jaxbContext.createUnmarshaller();
AddressesDto addressesDto = (AddressesDto) unmarshaller.unmarshal(bfr);
```

addresses.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<addresses>
    <address country="Bulgaria">
        <city>Sofia</city>
    </address>
    <address country="Spain">
        <city>Barcelona</city>
    </address>
</addresses>
```

Summary

SoftUni Foundation

- XML is another way to transfer data besides JSON
- XML document's format consists of **mark-up** and **content** elements
- JAXB is a library which helps us to read XML files and parse them to Java objects

Questions?

SoftUni

Software University

SoftUni Svetlina

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

SoftUni Diamond Partners

SoftUni Foundation

XSsoftware

SBTech
we know sports

telenor

SoftwareGroup
doing it right

NETPEAK

SmartIT

Postbank
Решения за твоето утре

SUPER HOSTING .BG

INDEAVR
Serving the high achievers

INFRAGISTICS

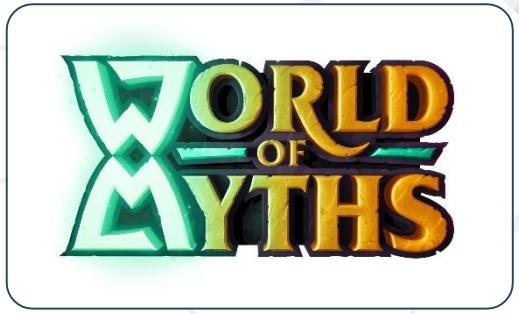
LIEBHERR

æternity

codeXio

SoftUni Organizational Partners

SoftUni Foundation



Trainings @ Software University (SoftUni)

SoftUni Foundation

- Software University – High-Quality Education and Employment Opportunities
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license