esprit Ecole Supérieure Privée d'Ingénierie et de Technologies	Année Universitaire : 2020-2021 **Ateliers SOA** **UP- WEB**

Atelier

Génération d'un document XML, XSD et java via JAXB (Marshalling et Unmarshalling)

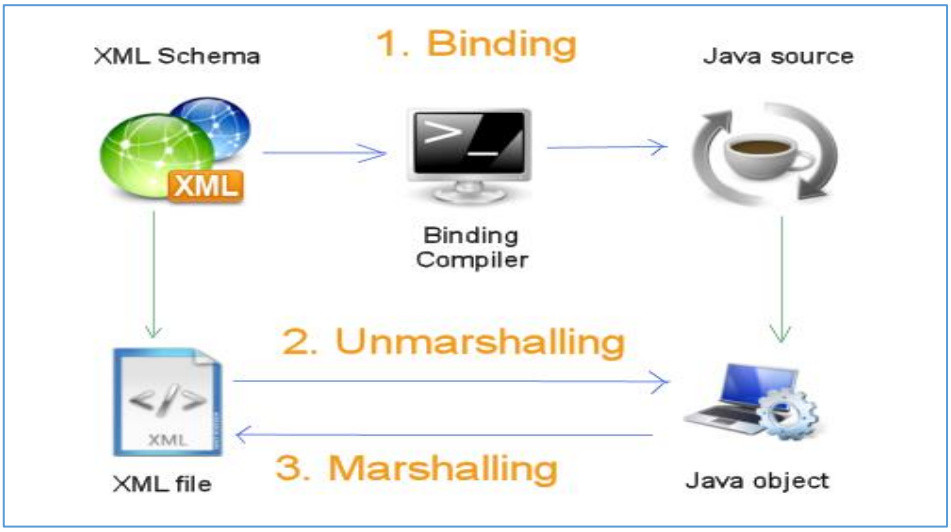
Objectifs

Générer un fichier XSD à partir d'une classe java
Générer une classe java à partir d'un fichier XSD
Générer un fichier XML qui correspond à une classe annotée via JAXB
Générer une instance java à partir d'un fichier XML

La mise en œuvre de JAXB 2.0

JAX-B 2.0 est une spécification qui permet de mapper des objets Java dans un document XML et vice versa au moyen d'opérations de sérialisation/désérialisation nommées marshalling/unmarshalling. Il permet aussi de générer des classes Java à partir d'un schéma XML et inversement.

L'API JAX-B est contenue dans la package **javax.xml.bind**



1. Création de la classe

a. Commencez par créer un projet de type **Maven** et ajoutez les propriétés suivantes dans le fichier **pom.xml** à l'intérieur de la balise **<Project>.**

```
<properties>
	<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
	<maven.compiler.source>1.8</maven.compiler.source>
	<maven.compiler.target>1.8</maven.compiler.target>
</properties>
```

b. Appuyer sur : **bouton droit sur projet > Maven > Update Project**

c. Créez la classe **Message.java** comme suit :

```
@XmlRootElement
Public class Message {
	private String from;
	private String to;
	private String text;
	private boolean isNew;
	//getters et setters
}
```

d. Générer les getters et setters ainsi que les constructeurs.

2. Génération d'un fichier XSD à partir de la classe Message

a. Placez-vous sous le package qui contient la classe Message puis exécutez la commande

« ***schemagen Message.java*** ».

Un fichier XSD qui porte le nom « **schema1.xsd** » va être généré sous le package ou vous avez exécuté la commande.

b. Analysez le fichier et déduisez les types XSD qui correspondent à chaque attribut de la classe.

❖ Personnalisation des entités

c. Ajoutez les annotations JAXB suivante à la classe « Message ».

```
@XmlRootElement
@XmlType(propOrder={"to","text","from"})
@XmlAccessorType(XmlAccessType.PROPERTY)
Public class Message {
	private String from;
	private String to;
	private String text;
	private Boolean isNew;

	@XmlElement(name="emetteur",required=true)
	public String getFrom() {
		return from;
	}
```

```
        Public void setFrom(String from) {
                this.from = from;
        }
        @XmlElement(name="destinataire")
        public String getTo() {
                return to;
        }
        Public void setTo(String to) {
                this.to = to;
        }
       @XmlAttribute
        public String getText() {
                return text;
        }
        Public void setText(String text) {
                this.text = text;
        }
        @XmlTransient
        Public Boolean isNew() {
                Return isNew;
        }
        Public void setNew(Boolean isNew) {
                this.isNew = isNew;
        }
}
```

Les annotations utilisées sont :

- **@XmlRootElement** : Associer la classe à l'élément racine du document XML.
- **@XmlType(propOrder)** : Définir l'ordre d'apparition des éléments dans le document XML
- **@XmlELement** : Convertir un attribut en un élément dans le document XML
- **@XmlAttribute** : Convertir un attribut en un attribut dans le document XML
- **@XmlTransiant** : Empêcher la sérialisation d'un attribut
- **@XmlAccessoryType** : Spécifie si les annotation JAXB seront situées sur les champs (field) ou sur les getters (property)

d. Analysez l'impact des annotations ajoutées sur le schema produit

3. Génération d'une classe Message à partir du fichier XSD

a. Placez-vous sous le dossier qui contient le fichier XSD et exécutez la commande « ***xjc schema1.xsd*** ». Un dossier nommé **generated** va être créé. Il contient la classe Message générée.

b. Analysez le contenu de la classe et essayez de comprendre le sens des annotations JAXB générées.

4. Génération du fichier xml à partir d'une classe java (Marshalling)

a. Créez la classe suivante qui permet de générer un fichier xml à partir de la classe Message
b. Utiliser la classe Marshaller de l'API JAXB pour générer le document XML à partir des objets déjà créés.

```
import java.io.File;
import java.util.ArrayList;
import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Marshaller;
import service.mail.ws.Message;

public class GenererMessageXml {
	public static void main(String[] args) throws JAXBException {
		Message message = new Message();
		message.setFrom("mohamed");
		message.setNew(true);
		message.setText("hello");
		message.setTo("ali");
		// création d'un contexte JAXB sur la classe Message
		JAXBContext context = JAXBContext.newInstance(Message.class);

		// création d'un marshaller à partir de ce contexte
		Marshaller marshaller = context.createMarshaller();

		// on choisit UTF-8 pour encoder ce fichier
		marshaller.setProperty("jaxb.encoding", "UTF-8");

		// et l'on demande à JAXB de formatter ce fichier de façon
		// à pouvoir le lire à l'oeil nu
		marshaller.setProperty("jaxb.formatted.output", true);

		// écriture finale du document XML dans un fichier message.xml
		marshaller.marshal(message, new File("message.xml"));
	}
}
```

Le fichier **XML** généré sera situé sous la **racine du projet**.

5. Génération d'une instance java à partir d'un document xml (Unmarshalling)

a. Créez et exécutez la méthode main suivante :

```
Public static void main(String[] args)  throws JAXBException {
   // création d'un contexte JAXB sur la classe Message
   JAXBContext context = JAXBContext.newInstance(Message.class) ;
  // création d'un unmarshaller
  Unmarshaller unmarshaller = context.createUnmarshaller() ;
  Message message= (Message)unmarshaller.unmarshal(new File("message.xml")) ;
  System.out.println("From = " + message.getFrom()) ;
  System.out.println("To = " + message.getTo()) ;
  System.out.println("Text = " + message.getText()) ;
}
```