




# JAX-B

**Module SOA**  
**A.U 2019-2020**



# Objectifs

- 
- **Comprendre le principe de JAX-B**
  - **Apprendre l'utilisation de JAX-B**



# Plan



- Présentation de JAX-B
- Fonctionnement
- Cas d'usages
- Avantages
- Mapping des types de données
- Utilisation
- Annotations



# Présentation de JAX-B

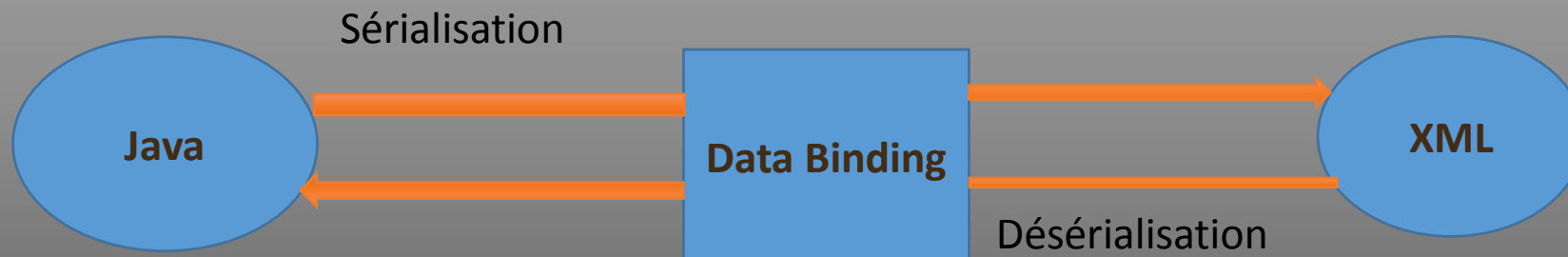


- ❖ Java Architecture for XML Binding
- ❖ JAX-B est un API Java permettant la gestion de données XML.
- ❖ JAX-B 2.0 est incorporée dans Java SE 6
- ❖ JAX-B permet plus particulièrement l'utilisation du "Data Binding"

# Présentation de JAX-B

Qu'est ce que le « Data Binding » ou association de données ?

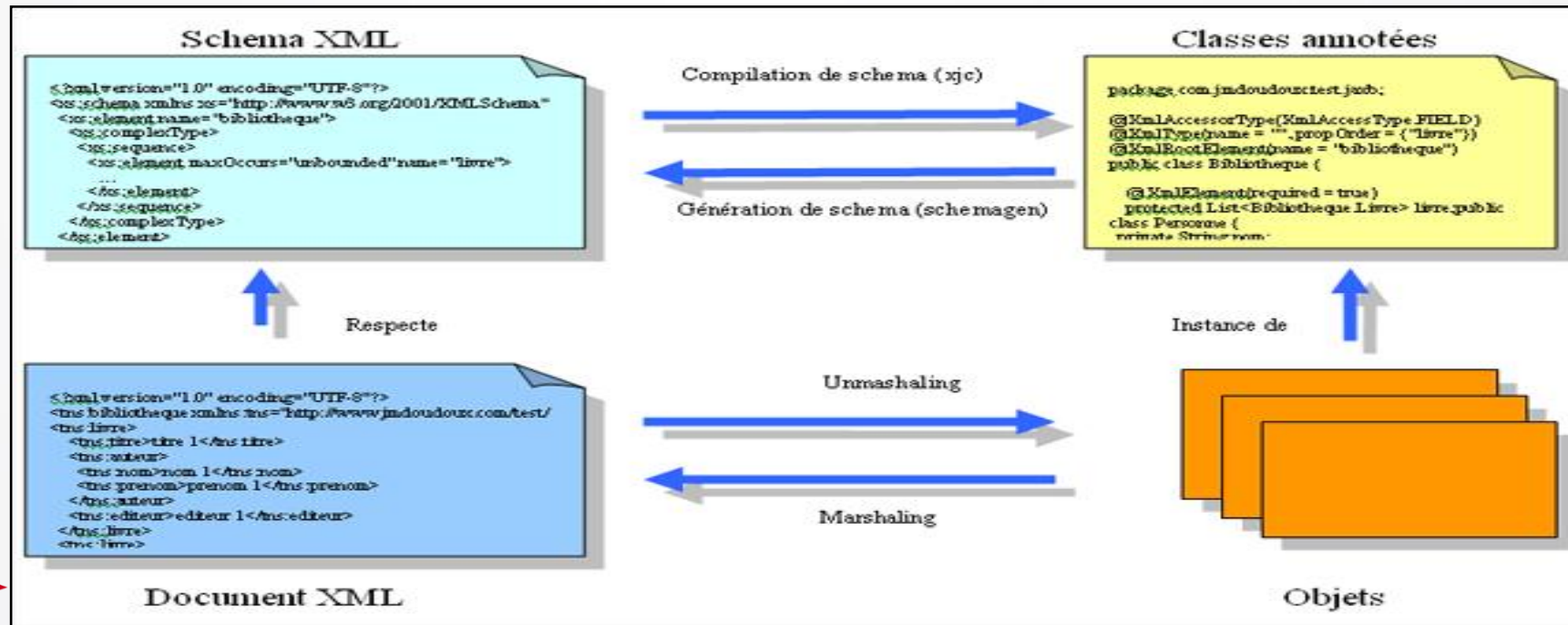
Le Data Binding est une technologie permettant d'automatiser la transformation des fichiers XML en objets Java et inversement.



# Fonctionnement

JAX-B permet deux types de transformations :

- ❖ Classes java ↔ Schéma XML
- ❖ Instances de classes java ↔ XML



# Cas d'usages

JAX-B est utilisé par :

❖ JAX-WS

Les applications JAX-WS utilisent JAX-B pour la conversion de données entre les classes Java et, WSDL et SOAP.

❖ JAX-RS

Les services web RESTful utilisent JAX-B pour la transformation des données échangées en XML



# Avantages



- ❖ Génération de classes automatisée : Gain de temps pour l'utilisateur
- ❖ Meilleure intégrité des données : JAX-B comporte des outils chargés de vérifier l'intégrité des données reçues.  
Lorsqu'une erreur intervient, le système pourra lever des exceptions
- ❖ Validation des documents XML
- ❖ Gestion de la persistance de données stockées sous format XML : La redistribution des données consiste à récupérer le contenu de chaque instance de classe et de les insérer dans les fichiers XML



# Mapping des types de données

Types de données prédéfinis:

Types Java	Types XSD
boolean	boolean
float	float
int	int
String	string
Object	anyType

Liste complète :

[http://docs.oracle.com/cd/E13222\\_01/wls/docs103/webserv/data\\_types.html#wp223908](http://docs.oracle.com/cd/E13222_01/wls/docs103/webserv/data_types.html#wp223908)

# Mapping des types de données

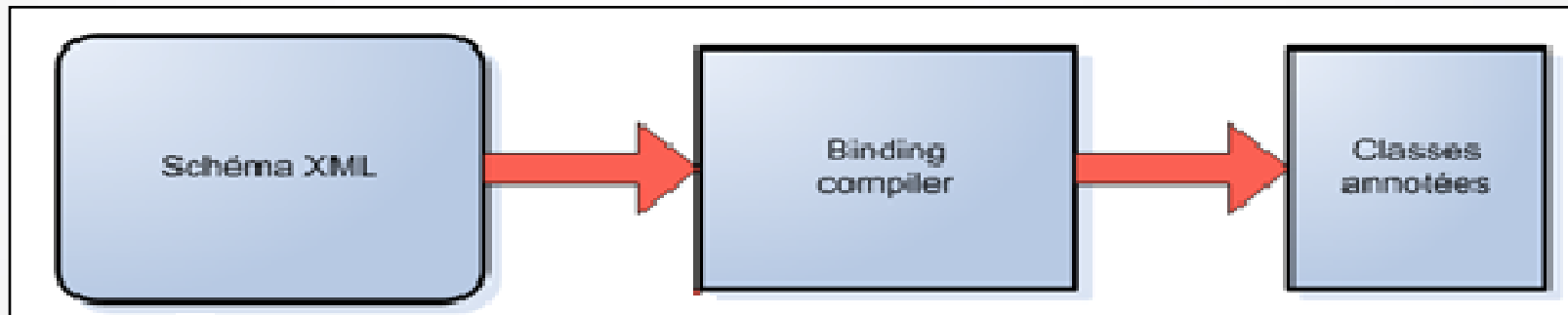
Types de données définis par l'utilisateur:

Types Java	Types XSD
Javabeau	<xsd:complexType>
Variable de Javabeau	Un élément sous <xsd:complexType>
Variable de Javabeau de type List	Un element sous <xsd:complexType> avec l'attribut maxOccurs="unbounded".

# Utilisation

## Génération des classes à partir d'un schéma

- ❖ L'outil **xjc** permet de **générer les classes** à partir d'un schéma XML
- ❖ Exemple: *xjc personne.xsd*
- ❖ Les classes générées :
  - **Personnes.java** : classe qui encapsule le document XML
  - **ObjectFactory.java** : fabrique qui permet d'instancier la classe Personne



JAX-B

# Utilisation

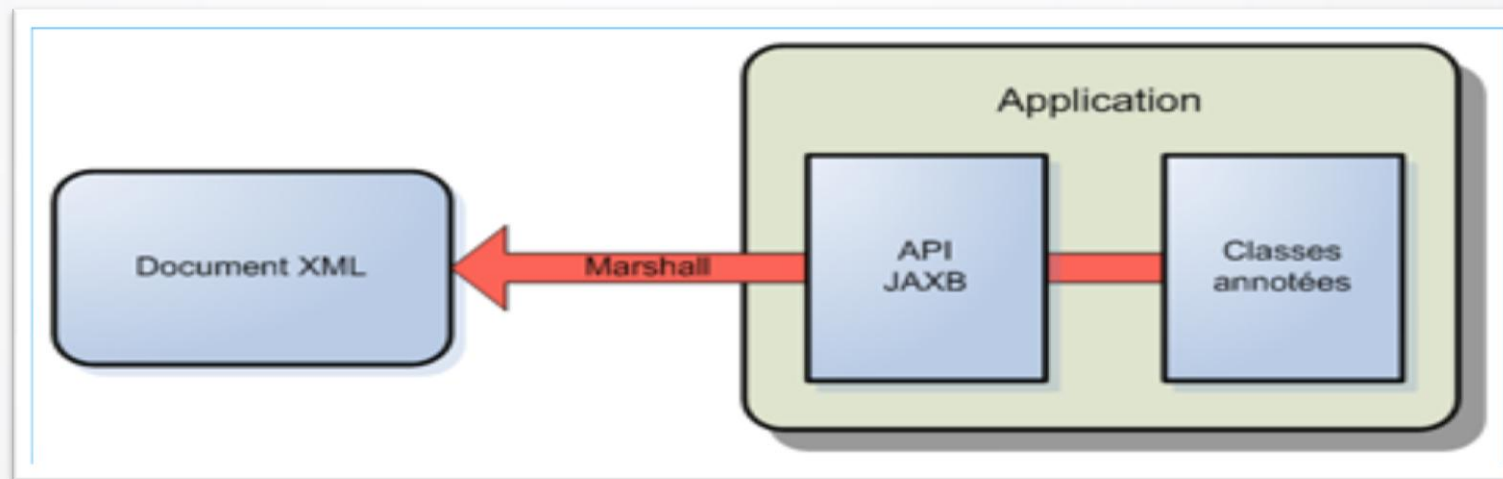
## Génération d'un schéma à partir d'une classe java

- ❖ L'outil **schemagen** permet de **générer le schéma XML** à partir d'une classe
- ❖ Exemple: *schemagen Personne.java*
- ❖ Les fichiers générées :
  - **Personne.class** : Bytecode de la classe Personne compilée
  - **Schema1.xsd** : fichier XSD correspondant

# Utilisation

## Génération d'un document XML à partir d'une instance Java (Sérialisation / Marshalling)

- 1) Ajouter les annotations nécessaires à la classe java
- 2) Utiliser la classe Marshaller de l'API JAX-B pour générer le document XML à partir des objets déjà créés.

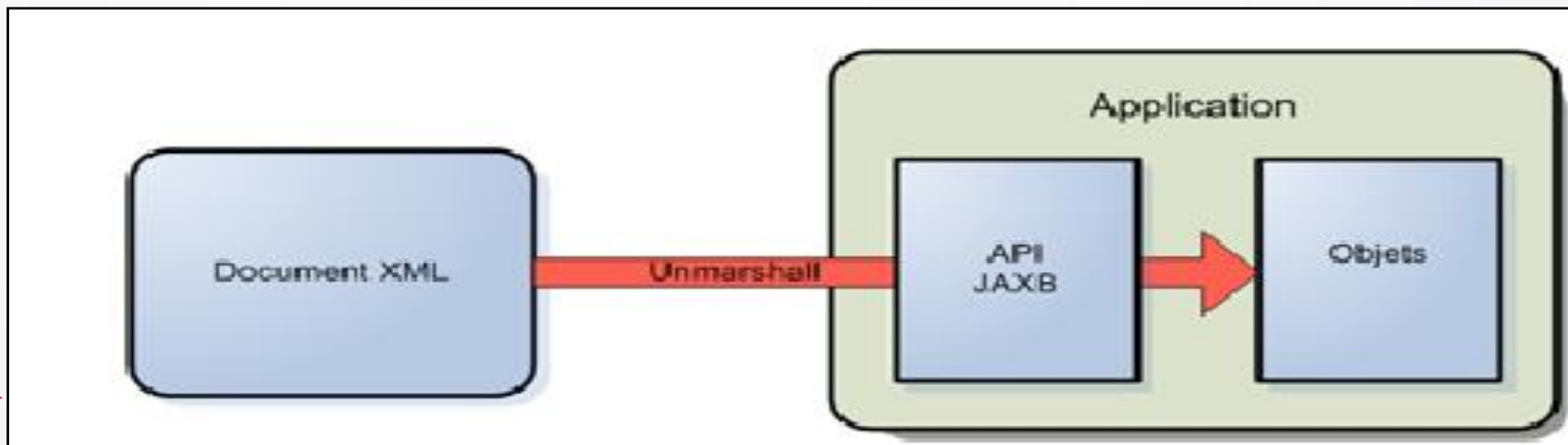


JAX-B

# Utilisation

## Mapping d'un document XML à des Objets (Desérialisation / Unmarshalling )

L'API JAX-B propose de transformer un document XML en un ensemble d'objets qui vont encapsuler les données et la hiérarchie du document. Ces objets sont des instances des classes générées à partir du schéma XML.





# Annotations JAX-B



- ❖ Les annotations JAX-B utilisées dans les classes java permettent la génération et la personnalisation:
  - ✓ Des schémas XSD générés
  - ✓ Des documents XML générés
- ❖ Les annotations JAX-B sont définies dans le package `javax.xml.bind.annotation`.



# Annotations JAX-B



Annotation	Description
<b>XmlRootElement</b>	Spécifier la racine du document XML.
<b>XmlElement</b>	Convertir une propriété de la classe en un élément dans le document XML.
<b>XmlAttribute</b>	Convertir une propriété de la classe en un attribut dans le document XML.
<b>XmlTransient</b>	Retire des éléments pris en compte pour la création des schémas et des documents XML.





# Annotations JAX-B



Annotation	Description
<b>XmlType</b>	Permet de fixer l'ordre dans lequel les champs de cette classe doivent être enregistrés dans le document XML.
<b>XmlAccessorType</b>	Contrôler l'ordre des attributs et des propriétés dans la classe.
<b>XmlSchema</b>	Associer un espace de noms à un paquetage.



# Annotations JAX-B



//Par défaut l'élément XML généré est le nom de la classe « Personne »

@XmlElement(name="maPersonne")

@XmlType(propOrder={"prenom", "nom"})

public class Personne {

private int age;

private String nom;

private String prenom;

@XmlElement // Par défaut

public String getNom() { return nom; }

@XmlElement // Par défaut

public String getPrenom() { return prenom; }

@XmlTransient

public int getAge() { return age; }

}

JAX-B



# En Résumé



- ❖ Outil puissant qui analyse un schéma XML et génère à partir de ce dernier un ensemble de classes qui vont encapsuler les traitements de manipulation du document.
- ❖ Fournir au développeur un moyen de manipuler un document XML sans connaître XML ou les technologies d'analyse.
- ❖ Manipulations au travers d'objets java.
- ❖ Ces classes sont utilisées pour faire correspondre le document XML dans des instances de ces classes et vice et versa : ces opérations se nomment  
respectivement unmarshalling et marshalling.

# En Résumé

## Schéma XML

```
3 <xs:schema version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema#">
4   <xs:element name="formes" type="formes" />
5   <xs:complexType name="formes">
6     <xs:sequence>
7       <xs:choice minOccurs="0" maxOccurs="unbounded">
8         <xs:element name="carré" type="carré" />
9         <xs:element name="cercle" type="cercle" />
10      </xs:choice>
11    </xs:sequence>
12  </xs:complexType>
```

## Compilation de schéma (xjc)

## Classes annotées et leurs sous-éléments

```
// Définition de l'élément @XmlElement et de ses sous-éléments
@XmlRootElement
public class Formes {
    @XmlElement({
        @XmlElement(name = "carré", type = Carré.class),
        @XmlElement(name = "cercle", type = Cercle.class)
    })
    private ArrayList<Forme> formes = new ArrayList<Forme>();
    public ArrayList<Forme> getFormes() { return formes; }
    public void ajoutForme(Forme forme) { formes.add(forme); }
    public void supprimerFormes() { formes.clear(); }
```

## Respecte

## Génération de schéma (schemagen)

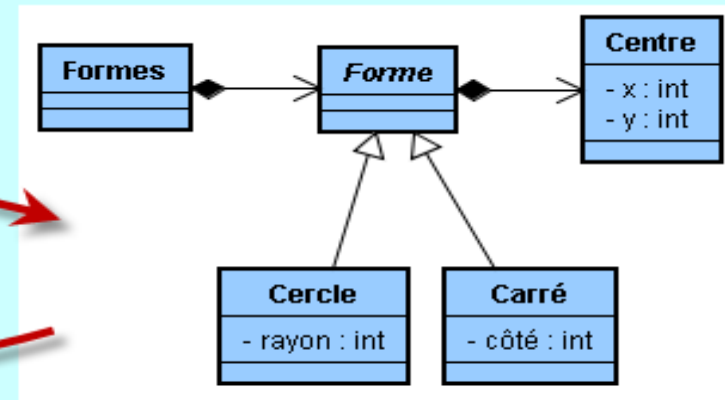
## Instances de

```
...xml GestionRépertoire.java Formes.xml Formes.java Principal.java
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <formes>
3   <cercle>
4     <centre x="100" y="73"/>
5     <rayon>25</rayon>
6   </cercle>
7   <carré>
8     <centre x="147" y="111"/>
9     <côté>100</côté>
10  </carré>
11  <cercle>
12    <centre x="196" y="92"/>
13    <rayon>50</rayon>
14  </cercle>
15 </formes>
```

## Document XML

## Unmarshalling

## Marshalling



## Objets issus des classes