JAXB (Java Architecture for XML Binding)

JAXB permet de générer un schéma XML à partir d'une classe Java, ou inversement.

Plusieurs types d'annotations :

- Sur un package
- Sur les classes et énumérations
- Sur les champs ou getters

Le tableau ci-dessous présente les principales annotations JAXB, où elles se posent ainsi que leur rôle :

Annotation	**Où ?**	**Rôle**
@XmlRootElement	Sur une classe	-Associe la classe annotée avec un nœud racine d'un document XML. -suffisante pour générer un document XML à partir d'une instance de cette classe.
@XmlType	Sur une classe	Le point principal est son attribut propOrder, qui permet de fixer l'ordre dans lequel les champs de cette classe doivent être enregistrés dans le document XML
@XmlAccessorType *(.FIELD / .PROPERTY /.PUBLIC /.NONE)*	Sur une classe	Indique quels champs d'une classe JAXB doit prendre en compte, on rajoute ainsi après l'annotation : • **.FIELD** : champs non statiques • **.PROPERTY** : getters/setters • **.PUBLIC** : getters/setters et champs publics non statiques • **.NONE** : aucun champ
@XmlEnum	Sur une classe	Précise comment les valeurs d'une énumération vont être écrites sur le code XML
@XmlElement	Sur un champ ou un getter	• Associe un champ ou un getter à un nœud d'un document XML • Spécifie le nom de cet élément

@XmlTransient	Sur un champ ou un getter	Retire le champ ou getter sur lequel elle est posée des éléments pris en compte pour la création des schémas et des documents XML.
@XmlElements	Sur un champ ou un getter	Peut spécifier que chaque élément d'une liste, du fait de sa classe, est associée à un nœud différent, si le champ annoté est de type *List*
@XmlList	Sur des champs ou getters de types List	Permet d'écrire une liste dans une unique élément XML, et les éléments séparés par des espaces
@XmlAttribute	Sur un champ ou un getter	Permet d'écrire le champ annoté dans un attribut XML plutôt que dans un sous-élément de l'élément XML parent
@XmlValue	Sur un champ ou un getter	Indique que les instances de cette classe sont représentées par une valeur simple unique, donnée par le champ qui porte cette annotation. Ainsi, lorsque ces objets seront utilisés ailleurs, ils seront représentés par cette valeur simple.

Syntaxe utilisée pour l'utilisation des annotations :

1) @XmlRootElement

```
//Sans nom de l'attribut
@XmlRootElement

//Avec nom de l'attribut
@XmlRootElement(name= '' nom_classe '')
```

2) @XmlType

```
@XmlType(propOrder = ''attribut1, attribut2, … '')
```

3) @XmlAccesorType

```
@XmlRootElement(name = "classe")
@XmlAccessorType(XmlAccessType.FIELD)
```

4) @XmlEnum

```
@XmlEnum(String.class)
public enum enumeration {

      enum1, enum2, enum3, enum4
}
```

5) @XmlElement

```
@XmlElement(name = « champs en question »)
```

Où on ajoute @XmlElement avant l'attribut concerné :

```
@XmlElement
private String firstName ;
```

6) @XmlTransient

On ajoute @XmlTransient avant l'attribut concerné :

```
@XmlTransient
private int num ;
```

7) @XmlList

On ajoute @XmlList avant de définir la liste :

```
@XmlList
      Private List<String> liste1
```

8) @XmlAttribute

On ajoute @XmlAttribute avant de définir l'attribut en question :

```
@XmlAttribute
private Integer id ;
```

9) @XmlValue

```
@XmlValue
private int montant ;
```