

# VI. XML SCHEMA

...

Validation en ligne <http://xmlvalidation.com>

# Schéma XML

## Apports schémas /DTD

- Un grand nombre de types de données intégrées comme les booléens, les entiers, les intervalles de temps, etc. De plus, il est possible de créer de nouveaux types par ajout de contraintes sur un type existant.
- Le support des espaces de nom.
- La notion d'héritage: les éléments peuvent hériter du contenu et des attributs d'un autre élément.
- Les contraintes sur les éléments

# Plan de cette partie

1. Schéma XML
2. Lier un schéma XML à un document XML
3. Contenu d'un schéma XML
  - 3.1. Déclaration d'éléments
  - 3.2. Déclaration d'attributs
  - 3.3. Types de base et Types simples
  - 3.4. Types complexes
  - 3.5. Dérivation de nouveaux types complexes
    - a) Dérivation par extension
    - b) Dérivation par restriction
  - 3.6. Groupes d'éléments et d'attributs
  - 3.7. Clés et références de clés

- Présentation
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

# Schéma XML?

- Développé par le W3C
- Norme officielle
- Syntaxe XML
- Exploitation des espaces de noms
- Modèle riche et évolué
- Mais verbeux et complexe
- La version 1.0 est proposée par le consortium W3C en 2001 et une révision en 2004 et la version 1.1 est une recommandation du W3C depuis avril 2012.

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- [Présentation](#)
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

## • Document XML : livre.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<livre">
  <auteurs>
    <auteur nom="Martin" prenom="Bill" />
    <auteur nom="Bob" prenom="Bobby"/>
  </auteurs>
  <sections>
    <section titre="Section1">
      <chapitre titre="un chapitre">
        <paragraphe>paragraphe 1 </paragraphe>
        <paragraphe>paragraphe 2 </paragraphe>
      </chapitre>
    </section>
  </sections>
</livre>
```

## • DTD pauvre

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT livre (auteurs, sections)>
<!ELEMENT auteur EMPTY>
<!ATTLIST auteur
          nom (Bob | Martin) #REQUIRED
          prenom (Bill | Bobby) #REQUIRED
>
<!ELEMENT auteurs (auteur+)>
<!ELEMENT section (chapitre)>
<!ATTLIST section
          titre (Section1 | Section2) #REQUIRED
>
<!ELEMENT chapitre (paragraphe+)>
<!ATTLIST chapitre
          titre CDATA #REQUIRED
>
<!ELEMENT sections (section+)>
<!ELEMENT paragraphe (#PCDATA)>
```

## Schéma XML

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Présentation
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

- Un fragment de schéma pour ce document : apprécier la richesse : **livre.xsd**

```
<?xml version="1.0" encoding="UTF-8"?>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="auteur">
      <xs:complexType>
        <xs:attribute name="nom" use="required">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="Bob"/>
              <xs:enumeration value="Martin"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="prenom" use="required">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="Bill"/>
              <xs:enumeration value="Bobby"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:complexType>
    </xs:element>
    .....
  </xs:schema>
```

## Schéma XML

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Présentation
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

# Structure d'un schéma XML

Est un document XML bien formé(arbre) avec :

- **Une racine schema** (xs:schema ou xsd:schema),
- **L'espace de noms des schémas XML** est identifié par l'URL <http://www.w3.org/2001/XMLSchema>. Généralement associé, au préfixe **xsd** ou à **xs**.
- Une suite de définitions de **types**, d'éléments et d'attributs

```
<?xml version=" 1.0 " encoding=" iso-8859-1 ">
<xsd:schema xmlns:xs= "http://www.w3.org/2001/XMLSchema">
  <!--déclarations d'éléments, d'attributs et définitions de types -->
  .....
</xsd:schema>
```

## Schéma XML

Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Présentation
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

# Espace de noms et schéma XML

- Espaces de noms sont spécifiés sous la forme d'une URL

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xs:schema      xmlns:xs="http://www.w3.org/2001/XMLSchema"
                 targetNamespace="http://www.lsis.org/sellamis/"
                 elementFormDefault="qualified"
                 attributeFormDefault="unqualified"
                 version="1.0">.....
</xs:schema>
```

- Et déclarés avec l'attribut `xmlns` et avec le préfixe `xs` ou `xsd`
- **targetNamespace**: spécifie un espace de noms cible dans lequel les éléments sont définis.
- **elementFormDefault** et **attributeFormDefault**: définir dans un même schéma des attributs et éléments qualifiés (c'est à dire appartenant à un espace de noms) et non qualifiés (sans espace de noms).



## Schéma XML

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Présentation
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

# Lier un schéma XML à un document XML

- Un document XML décrit par un XSD est appelé *instance document*.
- Dans le document XML: Il faut déclarer le namespace **xsi:** (XMLSchema-instance)
- 2 façons de lier un schéma XSD à un document XML:
  - Sans espace de noms
  - Avec espace de noms

## Schéma XML

Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Présentation
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

- Sans espace de noms

- Le document XML doit inclure :

- Dans la balise ouvrante de l'élément racine:  
`xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"`
    - L'attribut *xsi:noNamespaceSchemaLocation* définit l'URL de votre unique XSD permettant de valider l'intégralité du document.

```
<livre xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
      xsi:noNamespaceSchemaLocation="livre.xsd">
```

## Schéma XML

Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Présentation
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

## • Avec espace de noms

### ○ Le document XML doit inclure :

- une déclaration pour le XMLSchema-instance namespace
- un attribut **xsi:schemaLocation** qui dit où trouver XSD
- Cet attribut peut contenir plusieurs **pairs** "namespace-URL "

```
<livre xmlns="http://www.livre.org"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.livre.org livre.xsd">
```

### ○ Le schéma XML (livre.xsd)

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.livre.org"
            xmlns="http://www.livre.org"
            elementFormDefault="qualified">
```

## Schéma XML

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Présentation
- Espace de noms et schéma XML
- Lier un schéma XML à un document XML
- Contenu d'un schéma XML

## Qu'est ce qu'on déclare dans un schéma XML?

- Des éléments
- Des attributs
- **Leurs types:**
  - Simples pour les attributs
  - Simples ou complexes pour les éléments
  - Qui peuvent être définis par extension ou par restriction
- Des groupes d'éléments et d'attributs

Schéma XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- À partir d'un type local
- Référence à une autre définition

# Déclaration d'éléments <xs:element/>

## a) À partir d'un type local

```
<xs:element
  default = string
  final = (#all | List of (extension | restriction))
  fixed = string
  name = NCName
  type = QName
  {any attributes with non-schema Namespace}...>
  Content: (xs:annotation?, ((xs:simpleType | xs:complexType)?, (unique | key | keyref)*))
  ...
</xs:element>
```

- **Éléments parents:** schema, choice, all, sequence

Schéma XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- À partir d'un type local
- Référence à une autre définition

## Comment on fait référence à un élément global?

### b) Référence à une autre définition

```
<xs:element  
  |   ref = QName  
  |  
  |  
  |  
</xs:element>
```

Les attributs **Name** et **ref** ne peuvent pas être présents en même temps

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Définition d'un attribut
- Référence à une autre définition
- Indicateurs d'occurrence

# Déclaration d'attributs

- Déclaration: `<xs:attribute/>`

```

<xs:attribute
  default = string
  fixed = string
  form = (qualified | unqualified)
  name = NCName
  type = QName
  use = (optional | prohibited | required): optional
  {any attributes with non-schema Namespace...}
  ...>
Content: (xs:annotation?, (xs:simpleType?))
</xs:attribute>

```

- **Éléments parents:** attributeGroup, schema, complexType, restriction (simpleContent), extension (simpleContent), restriction (complexContent), extension (complexContent)

Déclaration des éléments

**Déclaration des attributs**

Types de base

Types simples

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Définition d'un attribut
- Référence à une autre définition
- Indicateurs d'occurrence

## Déclaration d'attributs

- **Faire Référence à un attribut déjà défini**

```
<xs:attribute  
  ref = QName  
  .....autres attributs optionnels .....  
</xs:attribute>
```

La référence doit être locale (nichée dans un type complexe)



Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Définition d'un attribut
- Référence à une autre définition
- Indicateurs d'occurrence

Comment indiquer qu'un attribut est **optionnel**?

**Attribut Use:** dire si l'attribut est optionnel (optional), obligatoire (required) ou interdit (prohibited)

```
<xs:attribute name="lang" type="xs:string" use=" optional"/>
```

Par défaut un attribut est optionnel

Comment préciser qu'un attribut a une valeur **par défaut**?

- ```
<xs:attribute name="lang" type="xs:string" default="EN"/>
```

Comment on peut affecter une **valeur fixe** à un attribut?

- ```
<xs:attribute name="lang" type="xs:string" fixed="EN"/>
```

Schéma XML  
 Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
 Types complexes  
 Dérivation de types complexes  
 Groupes  
 Clés et références

- Définition d'un attribut
- Référence à une autre définition
- Indicateurs d'occurrence

Quels types de données sont déclarés dans un schéma XML?

- Les schémas XML permettent de définir deux types de données:  
**simple et complexe.**

		Types Simples	Types complexes
<b>Attributs</b>		X	
<b>Contenu de l'Élément</b>	Vide+texte (PCDATA)	X	
	Attribut		X
	Autres éléments (élément composé)		X
	Texte+attributs		X
	Éléments+attributs +texte (contenu mixte)		X

Schéma XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Définition d'un attribut
- Référence à une autre définition
- Indicateurs d'occurrence

## Quels types de données sont déclarés dans un schéma XML?

- Les schémas XML permettent de définir deux types de données:
  - a) **Types simples:**
    - Utilisés pour les déclarations d'attributs, d'éléments dont le contenu se limite à des données atomiques.
    - **Ne peuvent pas contenir** d'autres éléments ou attributs
    - **2 catégories**
      - *Types de base*: entier, chaîne de caractères, etc.
      - À définir
  - b) **Types complexes:**
    - Type **composé** d'autres éléments ou contenant des attributs
    - Permet de définir des séquences d'éléments, des ensembles, des cardinalités, etc

# Plan de cette partie

1. Schéma XML
2. Lier un schéma XML à un document XML
3. Contenu d'un schéma XML
  - 3.1. Déclaration d'éléments
  - 3.2. Déclaration d'attributs
  - 3.3. Types de base et Types simples
  - 3.5. Types complexes
  - 3.5. Dérivation de nouveaux types complexes
    - a) Dérivation par extension
    - b) Dérivation par restriction
  - 3.6. Groupes d'éléments et d'attributs
  - 3.7. Clés et références de clés

- Les chaînes et les données
- Les décimaux et les durées
- Les entiers
- Les noms
- Hiérarchie de types

# Types de base

- Les chaînes et les données
- Les décimaux et les durées
- Les entiers
- Les noms

# Types de base

## ○ Les chaînes:

- **xs:string**: toute chaîne de caractères
- **xs:normalizedString**: dérive du type String mais avec une suite de blancs compactée (ne contient pas de tabulation, saut de ligne ou retour chariot)
- **xs:token**: idem que normalizedString mais ne contenant pas des espaces en début ou en fin ou des espaces consécutifs.

## ○ Les données binaires

- xs:base64Binary
- xs:hexBinary

Schema XML  
Déclaration des éléments  
Déclaration des attributs  
**Types de base**  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Les chaînes et les données
- Les décimaux et les durées
- Les entiers
- Les noms
- Hiérarchie de types

# Types de base

## ○ Les décimaux:

- xs:decimal
- xs:float et xs:double avec les infinis

## ○ Les durées

- xs:time (HH:MM:SS.sss)
- xs:date (YYYY-MM-DD)
- xs:dateTime (dateTime)

Schema XML  
Déclaration des éléments  
Déclaration des attributs  
**Types de base**  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Les chaînes et les données
- Les décimaux et les durées
- Les entiers
- Les noms
- Hiérarchie de types

## ○ Les entiers

Taille non fixée :

- xs:integer
- xs:positiveInteger: nombre entier strictement positif
- xs:negativeInteger: nombre entier strictement négatif
- xs:nonNegativeInteger: nombre entier positif ou nul
- xs:nonPositiveInteger: nombre entier négatif ou nul

Taille définie :

- xs:int et xs:unsignedInt (32 bits)
- xs:long et xs:unsignedLong (64 bits)
- xs:short et xs:unsignedShort (16 bits)
- xs:byte et xs:unsignedByte (8 bits)
- xs:boolean (true, false, 0, 1)



Schema XML  
Déclaration des éléments  
Déclaration des attributs  
**Types de base**  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Les chaînes et les données
- Les décimaux et les durées
- Les entiers
- Les noms
- Hiérarchie de types

## ○ Les noms:

- xs:Name nom simple
- xs:NCName nom simple sans « : »
- xs:QName nom qualifié
- xs:anyURI

## ○ Les clefs et les références

- xs:ID
- xs:IDREF
- xs:IDREFS

## ○ Les données de la DTD

- xs:ENTITY
- xs:ENTITIES
- xs:NOTATION
- xs:NMTOKEN
- xs:NMTOKENS

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

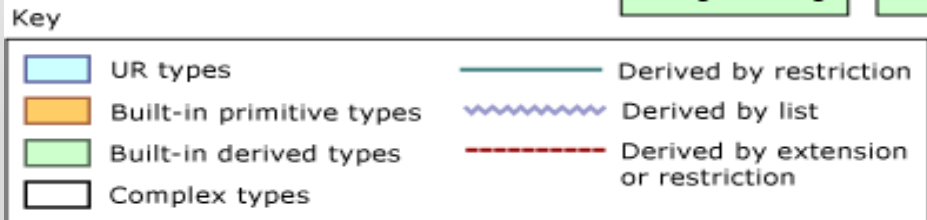
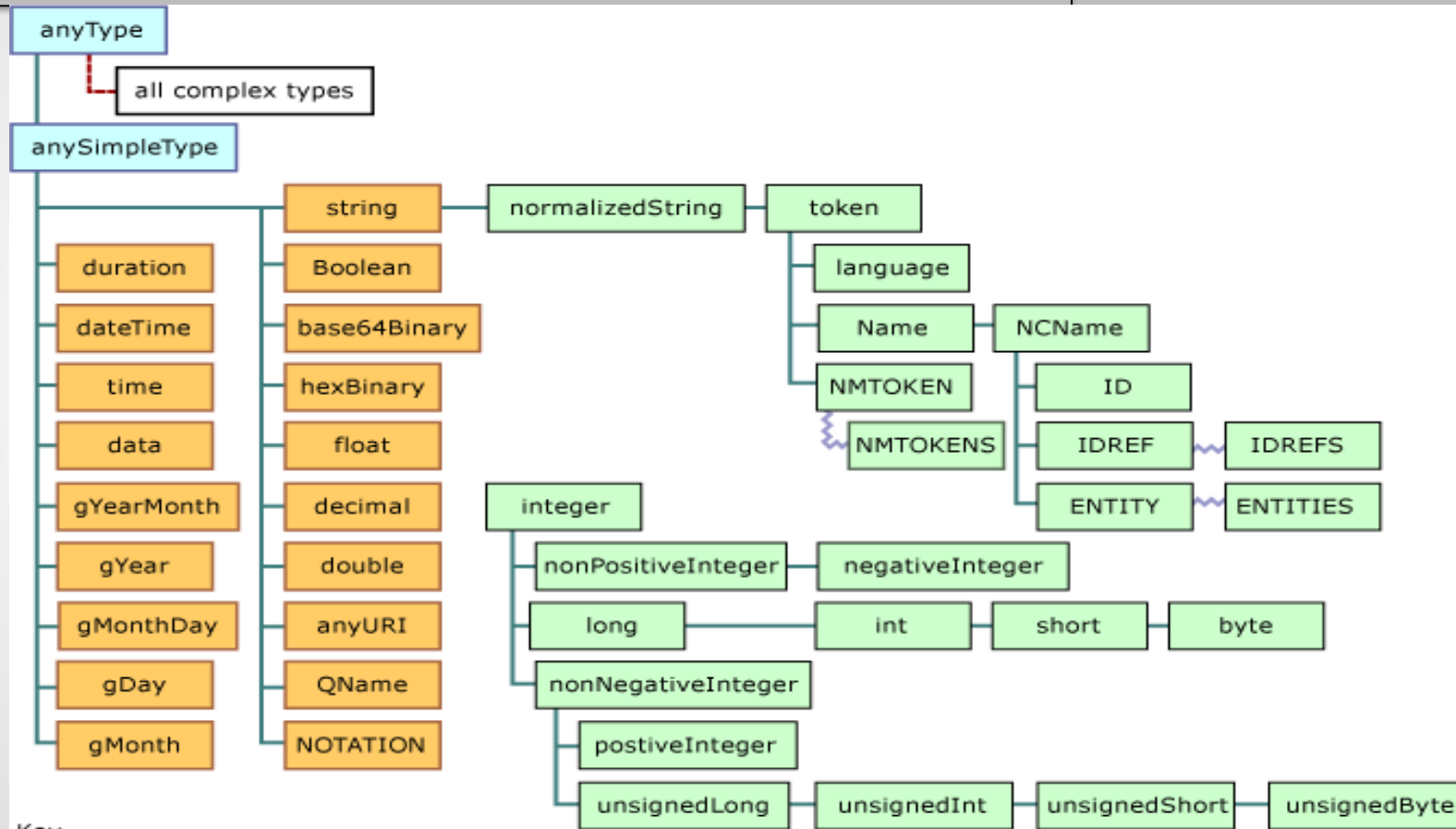
Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Les chaînes et les données
- Les décimaux et les durées
- Les entiers
- Les noms
- Hiérarchie de types



Source: <https://msdn.microsoft.com/fr-fr/library/bb399504.aspx>

Déclaration des éléments

Déclaration des attributs

Types de base

**Types simples**

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Dérivation par restriction
- Types listes
- Types union

## Comment peut on définir un type simple?

**Syntaxe:** `<xs:simpleType>`

```
<xs:simpleType  
  final = (#all | (list | union | restriction))  
  id = ID  
  name = NCName  
  {any attributes with non-schema Namespace}...>  
Content: (xs:annotation?, (xs:restriction | xs:list | xs:union))  
</xs:simpleType>
```

Peut être

- **Nommé:** défini directement sous la racine et muni de l'attribut *name* qui donne son nom. `<xs:simpleType name="monType">`
- **Anonyme** `<xs:simpleType>` sera inclus dans les déclarations `xs:element` ou `xs:attribute` des éléments ou attributs qu'il sert à typer.

## Définir des Types simples

**1) Dérivation des types par restriction <xs:restriction/>**  
(application d'une facette): créer un sous type en limitant certaines caractéristiques (e.g: longueur d'une chaîne)

**2) Définition des types listes <xs:list/>**: ensemble de valeurs de même type séparés par un blanc. (possibilité d'appliquer une restriction sur une liste)

**3) Définition des types union:** sert à autoriser plusieurs types.

## Dérivation par restriction: <xs:restriction/>

- Permet d'obtenir un type dérivé à partir d'un type de base ou défini
- Syntaxe:

```
<xs:restriction base= "...">  
  
</xs:restriction>
```
- L'attribut **base** donne le nom du type de base. Celui-ci peut être un type prédéfini ou un type défini dans le schéma
- **Exemple:** créer un type simple, appelé MonEntier, limité aux valeurs comprises entre 0 et 99 inclus. On dérive ce type à partir du type simple prédéfini nonNegativeInteger, en utilisant la facette maxExclusive.

```
<xs:simpleType name= " monEntier ">  
  <xs:restriction base= "xs:nonNegativeInteger">  
    <xs:maxExclusive value= "100 "/>  
  </xs:restriction>  
</xs:simpleType>
```

Schema XML Déclaration des éléments Déclaration des attributs Types de base Types simples Types complexes Dérivation de types complexes Groupes Clés et références	<ul style="list-style-type: none"> <li>• Dérivation par restriction</li> <li>• Types listes</li> <li>• Types union</li> </ul>
--	---

## Dérivation par restriction : XML schema définit 12 facettes

	Facette	Description
Contraintes textuelles	Enumeration	Définit une liste de valeurs possibles.
	Length	Nombre exact de caractères ou d'item de liste autorisés ( $\geq 0$ )
	maxLength	Nombre maximum de caractères autorisés ou d'item de liste autorisés ( $\geq 0$ )
	minLength	Nombre minimum de caractères autorisés ou d'item de liste autorisés ( $\geq 0$ )
	WhiteSpace	Spécification de traitement des espaces
Expression régulière	Pattern	Définit la séquence de caractères acceptables en fonction d'une expression régulière.
Contraintes numériques	fractionDigits	Spécifie le nombre maximal de décimales à droite du point décimal
	maxExclusive	Spécifie (exclusivement) la valeur maximum légale pour le type
	maxInclusive	Spécifie (inclusivement) la valeur maximum légale pour le type
	minExclusive	Spécifie (exclusivement) la valeur minimum légale pour le type
	minInclusive	Spécifie (inclusivement) la valeur minimum légale pour le type
	totalDigits	Spécifie le nombre maximum de chiffres dans une valeur décimale

Schema XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
**Types simples**  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Dérivation par restriction
- Types listes
- Types union

## Dérivation par restriction

- *Restriction à un ensemble de valeurs et gestion des blancs*

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  <xs:element name="marque">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="dell"/>
        <xs:enumeration value="hp"/>
        <xs:enumeration value="asus"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
</xs:schema>
```

Restriction du type prédéfini **string**

Liste de 3 valeurs: dell, hp, asus

<marque>dell hp</marque>

Doc non valide  
(valeur non listée)

<marque>hp</marque>

Doc valide

<marque>asus </marque>

Doc non valide (attention  
aux espaces)

Schema XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
**Types simples**  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Dérivation par restriction
- Types listes
- Types union

## Dérivation par restriction

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  <xs:element name="password">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:minLength value="5"/>
        <xs:maxLength value="8"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
</xs:schema>
```

Entre 5 et 8  
caractères  
alphanumériques

<password> kmlnER </password>

<password>kmlnER </password>

Doc non valide (espaces non gérés)

Doc valide



Schema XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Dérivation par restriction
- Types listes
- Types union

## Dérivation par restriction

### ○ Restriction à une série de valeurs

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  <xs:element name="ref">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:pattern value="([a-z][A-Z]\d)+" />
        <xs:length value="6" />
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
</xs:schema>
```

Répétition du triplet (+)  
(Lettre minuscule,  
lettre majuscule, chiffre (\d))

6digits exactement

<ref>zA1tB5</ref>

Doc valide

<ref>xK7PL1</ref>

Doc non valide (minuscule,  
majuscule ensuite chiffre) et 6 caractères

Schema XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
**Types simples**  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Dérivation par restriction
- Types listes
- Types union

Comment peut on déclarer un ensemble de valeurs **de même type séparés** par un blanc?

### Définition de types listes: `<xs:list/>`

Un ensemble de valeurs de même type (des suites de types simples) séparées par un blanc.

```
<xs:simpleType>  
  <xs:list itemType="xs:int">  
  </xs:list>  
</xs:simpleType>
```

La liste 10 20 333 4 3 est donc bien cohérente avec ce type

- Il est également possible de créer une liste personnalisée, par « dérivation » de types existants.

```
<xs:simpleType name="listAges">  
  <xs:list itemType="age">  
  </xs:list>  
</xs:simpleType>
```

Exemple de texte valide: 10 34 67

Schema XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
**Types simples**  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

- Dérivation par restriction
- Types listes
- Types union

## Comment déclarer plusieurs types?

### Définition de type Union: `<xs:union/>`

- Sert à autoriser plusieurs types
- Les types simples sont spécifiés par leur nom (attribut `memberTypes`) ou leur définition (éléments `xs:simpleType`).
- **Exemple**

```
<xs:simpleType name= " numéroDeTéléphoneTechnique ">  
  <xs: union memberTypes= "xs:string  numéroDeTéléphone"/>  
</xs:simpleType>
```

- **Instance valide:**

```
<téléphone>18</téléphone>  
<téléphone>Pompiers </téléphone>
```

Déclaration des éléments

Déclaration des attributs

Types de base

**Types simples**

Types complexes

Dérivation de types complexes

Groupes

Clés et références

- Dérivation par restriction
- Types listes
- Types union

## Comment déclarer plusieurs types?

### Définition de type Union: `<xs:union/>`

```
<xs:simpleType name="Numerinconnu">
  <xs:union memberTypes="xs:integer ">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value='inconnu' />
      </xs:restriction>
    </xs:simpleType>
  </xs:union>
</xs:simpleType>
```

```
10
inconnu
20
```

# Plan de cette partie

1. Schéma XML
2. Lier un schéma XML à un document XML
3. Contenu d'un schéma XML
  - 3.1. Déclaration d'éléments
  - 3.2. Déclaration d'attributs
  - 3.3. Types de base et Types simples
  - 3.5. Types complexes
  - 3.5. Dérivation de nouveaux types complexes
    - a) Dérivation par extension
    - b) Dérivation par restriction
  - 3.6. Groupes d'éléments et d'attributs
  - 3.7. Clés et références de clés

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

**Types complexes**

Dérivation de types complexes

Groupes

Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Comment peut on définir un type complexe?

**Syntaxe: <xs:complexType/>**

```
<xs:complexType
  abstract = Boolean : false
  id = ID
  mixed = Boolean : false
  name = NCName
  {any attributes with non-schema Namespace...}....>
Content: (xs:annotation?, (xs:simpleContent | xs:complexContent | ((xs:group | xs:all |
xs:choice | xs:sequence)?, ((xs:attribute | xs:attributeGroup)*, xs:anyAttribute?))))
</xs:complexType>
```

Concerne le contenu **d'un élément**

Type lui-même composé d'autres éléments et/ou contenant des attributs.

Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
**Types complexes**  
Dérivation de types complexes  
Groupes  
Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

Quand est ce qu'on considère qu'un élément est de type complexe?

### 1. Éléments vides (contenant des attributs)

```
<product pid="1345"/>
```

### 2. Éléments qui contiennent d'autres éléments

```
<employee>  
<firstname>John</firstname>  
<lastname>Smith</lastname>  
</employee>
```

### 3. Éléments qui contiennent seulement du texte et des attributs

```
<food type="dessert">Ice cream</food>
```

### 4. Éléments qui contiennent d'autres éléments et du texte (Mixte)

```
<description> It happened on  
<date lang="norwegian">03.03.99</date> ...  
</description>
```

Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
**Types complexes**  
 Dérivation de types complexes  
 Groupes  
 Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## 1. Élément vide

- Ne peut contenir que des attributs
- **Définition d'un élément vide:**

```

<xs:element name="empty">
  <xs:complexType>
    <xs:attribute name="att" use="optional" type="xs:string">
    </xs:attribute>
  </xs:complexType>
</xs:element>
  
```

<empty/>

Doc valide

<empty att="un attribut"/>

Doc valide

<empty>55</empty>

Doc non valide



Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
**Types complexes**  
 Dérivation de types complexes  
 Groupes  
 Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## 1. Élément vide

```

<xs:element name="empty">
  <xs:complexType>
    <xs:attribute name="att" use="optional" type="xs:string">
    </xs:attribute>
  </xs:complexType>
</xs:element>

```

On peut également donner à l'élément de complexType un nom → **Définition d'un type global**

```

<xs:element name="empty" type="emptyType"/>

<xs:complexType name="emptyType">
  <xs:attribute name="att" use="optional" type="xs:string">
  </xs:attribute>
</xs:complexType>

```

→ si on emploie cette méthode, plusieurs éléments peuvent se rapporter au même type complexe • 43

## 2. Éléments composés d'éléments

- est un type complexe contenant un élément qui contient seulement d'autres éléments
- **Exemple** : élément "auteur" contenant d'autres éléments

```
<auteur>
  <nom> Smith </nom>
  <prenom> John</prenom>
</auteur>
```

- On peut définir l'élément "auteur" dans un schéma comme suit :

```
<xs:element name="auteur">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="nom" type="xs:string"/>
      <xs:element name="prenom" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

?

Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
**Types complexes**  
Dérivation de types complexes  
Groupes  
Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

### 3. Éléments qui contiennent seulement du texte et des attributs

- Met en commun 2 principes contradictoires: le contenu est du texte (type simple)+ un typage complexe puisque l'élément contient un attribut
- Se fait par **dérivation** → **une extension de type**
- **L'extension de type** est similaire à l'héritage des langages de POO. Elle permet de définir un nouveau type en ajoutant des éléments et/ou des attributs à un type.

Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
**Types complexes**  
Dérivation de types complexes  
Groupes  
Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

### 3. Éléments qui contiennent seulement du texte et des attributs

- Déclaration: **<xs:extension/>** (enfant de **xs:simpleContent**)

```
<xs:element name="auteur">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="nom" type="xs:string"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

- L'élément auteur est un type complexe dont le contenu simple (typé xs:string) a été étendu pour lui ajouter un attribut nom.

Déclaration des éléments
Déclaration des attributs
Types de base
Types simples
<b>Types complexes</b>
Dérivation de types complexes
Groupes
Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## 4. Types complexes à contenu mixte

- Peut contenir des sous éléments, des attributs et du texte
- Afin de spécifier qu'il peut contenir également du texte, on utilise l'**attribut *mixed*** de l'élément *xs:complexType*.
- Par défaut, *mixed*="false"; il faut dans ce cas forcer *mixed*="true". Par exemple:

```
<xs:element name="auteur">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element name="nom" type="xs:string"/>
      <xs:element name="prenom" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

- L'élément auteur:

```
<auteur>
  Professeur
  <nom> Smith </nom>
  <prenom> John</prenom>
</auteur>
```

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

**Types complexes**

Dérivation de types complexes

Groupes

Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Les indicateurs

- Permettent de contrôler l'usage des éléments à l'intérieur du type complexe.
- Les indicateurs:
  - **3 indicateurs d'ordre** (Order indicators)
  - **2 indicateurs de cardinalité ou occurrence** (Occurrence Indicators)

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

**Types complexes**

Dérivation de types complexes

Groupes

Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Les indicateurs d'ordre

- Sont employés pour définir comment les éléments devraient se produire
  - Sans ordre: (xs:All)
  - Les choix: (xs:Choice)
  - Les séquences (xs:sequence)

Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
**Types complexes**  
 Dérivation de types complexes  
 Groupes  
 Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Indicateur all: **<xs:all>**

- Les éléments cités doivent tous apparaître sans ordre précis
- Équivalent DTD: `<!ELEMENT auteur ((nom, prenom))|(prenom,nom))>`

```

<xs:element name="auteur">
  <xs:complexType>
    <xs:all>
      <xs:element name="nom" type="xs:string"/>
      <xs:element name="prenom" type="xs:string"/>
    </xs:all>
  </xs:complexType>
</xs:element>
  
```

```

<auteur>
  <nom> Smith </nom>
  <prenom>John</prenom>
</auteur>
  
```

Doc valide

```

<auteur>
  <prenom>John</prenom>
  <nom> Smith </nom>
</auteur>
  
```

Doc valide

```

<auteur>
  <prenom>John</prenom>
</auteur>
  
```

Doc invalide



Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
**Types complexes**  
 Dérivation de types complexes  
 Groupes  
 Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Indicateur Choice: **<xs:choice>**

- Un seul des éléments cités doit apparaître
- A les mêmes effets que l'opérateur | dans une DTD.

```
<xs:element name = "menu" >
  <xs:complexType>
    <xs:choice>
      <xs:element name= "entree" type= "xs:string" />
      <xs:element name= "dessert" type="xs:string"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
```

```
<menu>
<entree>Salade</entree>
</menu>
```

**Doc valide**

```
<menu>
<dessert>Pomme</dessert>
</menu>
```

**Doc valide**

```
<menu>
<entree>Salade</entree>
<dessert>Pomme</dessert>
</menu>
```

**Doc invalide**

Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
**Types complexes**  
 Dérivation de types complexes  
 Groupes  
 Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Indicateur Sequence <xs:sequence>

- Les éléments doivent tous apparaître dans l'ordre de leur déclaration

```

<xs:element name="auteur">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="nom" type="xs:string"/>
      <xs:element name="prenom" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
  
```

```

<auteur>
  <nom> Smith </nom>
  <prenom>John</prenom>
</auteur>
  
```

Doc valide

```

<auteur>
  <prenom>John</prenom>
</auteur>
  
```

Doc invalide

```

<auteur>
  <prenom>John</prenom>
  <nom> Smith </nom>
</auteur>
  
```

Doc invalide

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

**Types complexes**

Dérivation de types complexes

Groupes

Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Indicateur Sequence (suite)

```
<xs:element name = "plan">
  <xs:complexType>
    <xs:sequence>
      <xs:element name= "auteurs" type="xs:string"/>
      <xs:element name="chapitres" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

**OU**

```
<xs:complexType name = "planType">
  <xs:sequence>
    <xs:element name= "auteurs " type= "xs:string" />
    <xs:element name= "chapitres" type= "xs:string" />
  </xs:sequence>
</xs:complexType>
<xs:element name= "plan " type= "planType" >
```

L'élément plan contient nécessairement 2 éléments fils de contenu simple auteurs et chapitres.

**Équivalent de cet exemple avec une DTD**

```
<!ELEMENT plan (auteurs, chapitres)>
<!ELEMENT auteurs (#PCDATA)>
<!ELEMENT chapitres(#PCDATA)>
```

**Type global: planType**

Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
**Types complexes**  
Dérivation de types complexes  
Groupes  
Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Les indicateurs de cardinalité ou occurrence

- Spécifient le nombre d'occurrences d'un élément
- 2 indicateurs: maxOccurs et minOccurs
  - L'infini est caractérisé par la chaîne *unbounded*
- Parallèle avec la DTD:
  - ? : minOccurs= "0" maxOccurs= "1"
  - + : minOccurs= "1" maxOccurs= "unbounded"
  - \* : minOccurs= "0" maxOccurs= "unbounded"

Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
**Types complexes**  
 Dérivation de types complexes  
 Groupes  
 Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Indicateur maxOccurs

- Nombre maximal d'apparition d'un élément
- Les éléments cités ne doivent pas apparaître plus que cette cardinalité
- Le temps d'occurrence minimum par défaut est 1

```

<xs:element name = "child">
  <xs:complexType>
    <xs:sequence>
      <xs:element name= "name" type="xs:string"/>
      <xs:element name= "parent" type="xs:string" maxOccurs= "2" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
  
```

Doc invalide

```

<child>
  <name> Jean </name>
  <parent>John</parent>
  <parent>Zoe</parent>
</child>
  
```

Doc valide

```

<child>
  <name> Jean </name>
  <parent>John</parent>
  <parent>Zoe</parent>
  <parent>Jack</parent>
</child>
  
```

Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
**Types complexes**  
 Dérivation de types complexes  
 Groupes  
 Clés et références

- Définition d'un type complexe
- Éléments vides
- Éléments composés
- Extensions de type
- Éléments mixtes
- Indicateurs d'ordre
- Indicateurs d'occurrence

## Indicateur minOccurs

- Nombre minimal d'apparition d'un élément
- **Déclaration:** attribut **minOccurs**

```

<xs:element name = "parent">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="child" type="xs:string" minOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

<parent>
  <name> Jean </name>
  <child>John</child>
</parent>

```

Doc valide

```

<parent>
  <name> Jean </name>
</parent>

```

Doc invalide

# Plan de cette partie

1. Schéma XML
2. Lier un schéma XML à un document XML
3. Contenu d'un schéma XML
  - 3.1. Déclaration d'éléments
  - 3.2. Déclaration d'attributs
  - 3.3. Types de base et Types simples
  - 3.5. Types complexes
  - 3.5. Dérivation de nouveaux types complexes
    - a) Dérivation par extension
    - b) Dérivation par restriction
  - 3.6. Groupes d'éléments et d'attributs
  - 3.7. Clés et références de clés

- Dérivation par extension:
  - Extension d'un type simple
  - Extension d'un type complexe
- Dérivation par restriction: d'un type complexe

## Extension de Types complexes

**1) Type complexe à contenu Simple:** lui ajouter de nouveaux attributs.

- Similaire à l'extension de type simple (toujours `<xs:extension/>`)
- L'élément `<xs:extension>` est encore enfant d'un élément `<xs:simpleContent>`

**2) Type complexe à contenu Complexe:**

- Ajouter du contenu et/ou des attributs. Le contenu est ajouté après le contenu du type de base
- L'élément `<xs:extension>` est enfant d'un élément `<xs:complexContent>`



Schema XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
**Dérivation de types complexes**  
Groupes  
Clés et références

- Dérivation par extension:
  - Extension d'un type simple
  - Extension d'un type complexe
- Dérivation par restriction: d'un type complexe

## Extension de type simple

Déclaration: **<xs:extension/>** (enfant de **xs:simpleContent**)

```
<xs:element name="auteur">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="nom" type="xs:string"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

L'élément auteur est un type complexe dont le contenu simple (typé xs:string) a été étendu pour lui ajouter un attribut nom.

- Dérivation par extension:
  - Extension d'un type simple
  - Extension d'un type complexe
- Dérivation par restriction: d'un type complexe

## Extension de type complexe

```
<xs:complexType name="avecTitreType">  
  <xs:attribute name="titre" type="xs:string"/>  
</xs:complexType>
```

étendre

```
<xs:complexType name="sectionType">  
  <xs:complexContent>  
    <xs:extension base="avecTitreType">  
      <xs:sequence>  
        <xs:element maxOccurs="unbounded" minOccurs="2"  
          name="chapitre" type="chapitreType"/>  
      </xs:sequence>  
    </xs:extension>  
  </xs:complexContent>  
</xs:complexType>
```

Schema XML  
 Déclaration des éléments  
 Déclaration des attributs  
 Types de base  
 Types simples  
 Types complexes  
**Dérivation de types complexes**  
 Groupes  
 Clés et références

- Dérivation par extension.
  - Extension d'un type simple
  - Extension d'un type complexe
- Dérivation par restriction d'un type complexe

## Restriction de Types

- Définit des contraintes sur une définition **<xs:complexContent>** avec **<xs:restriction>**.

```

<xs:complexType name="personType">
  <xs:sequence>
    <xs:element name="nom" type="xs:string"/>
    <xs:element name="prenom" type="xs:string" minOccurs="0" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="employeType">
  <xs:complexContent>
    <xs:restriction base="personType">
      <xs:sequence>
        La restriction supprime l'élément prenom
        <xs:element name="nom" type="xs:string"/>
      </xs:sequence>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
  
```

# Plan de cette partie

1. Schéma XML
2. Lier un schéma XML à un document XML
3. Contenu d'un schéma XML
  - 3.1. Déclaration d'éléments
  - 3.2. Déclaration d'attributs
  - 3.3. Types de base et Types simples
  - 3.5. Types complexes
  - 3.5. Dérivation de nouveaux types complexes
    - a) Dérivation par extension
    - b) Dérivation par restriction
  - 3.6. Groupes d'éléments et d'attributs
  - 3.7. Clés et références de clés

Schema XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
**Groupes**  
Clés et références

- Groupe d'éléments
- Groupe d'attributs

## Groupe d'éléments: **<xs:group>**

- L'élément xs:group permet de regrouper un contenu complexe.

```
<xs:group  
id=ID  
name=NCName  
ref=QName  
maxOccurs=nonNegativeInteger|unbounded  
minOccurs=nonNegativeInteger  
any attributes  
>  
(annotation?,(all|choice|sequence)?)  
</xs:group>
```

Déclaration des éléments

Déclaration des attributs

Types de base

Types simples

Types complexes

Dérivation de types complexes

**Groupes**

Clés et références

- Groupe d'éléments
- Groupe d'attributs

## Groupe d'éléments: **<xs:group>**

### Exemple

```
<xs:group name="auteursGrp">  
<xs:sequence>  
<xs:element maxOccurs="unbounded" name="auteur" type="auteurType"/>  
</xs:sequence>  
</xs:group>
```

Schema XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
**Groupes**  
Clés et références

- Groupe d'éléments
- Groupe d'attributs

## Groupe d'attributs: **<xs:attributeGroup/>**

- Des définitions d'attributs communes à plusieurs définitions d'éléments peuvent être concentrées dans des groupes d'attributs.
- Les groupes d'attributs sont définis globalement utilisés par référence

```
<xs:attributeGroup
  id = ID
  name = NCName
  ref = QName
  {any attributes with non-schema Namespace...}>
  Content: (xs:annotation?), ((xs:attribute | xs:attributeGroup)*, xs:anyAttribute?)
</xs:attributeGroup>
```

## Groupe d'attributs

L'exemple suivant illustre un groupe d'attributs définis (**myAttributeGroup**) et utilisés dans un type complexe (**myElementType**).

```
<xs:attributeGroup name="myAttributeGroup">
  <xs:attribute name="someattribute1" type="xs:integer"/>
  <xs:attribute name="someattribute2" type="xs:string"/>
</xs:attributeGroup>

<xs:complexType name="myElementType">
  <xs:attributeGroup ref="myAttributeGroup"/>
</xs:complexType>
```



# Plan de cette partie

1. Schéma XML
2. Lier un schéma XML à un document XML
3. Contenu d'un schéma XML
  - 3.1. Déclaration d'éléments
  - 3.2. Déclaration d'attributs
  - 3.3. Types de base et Types simples
  - 3.5. Types complexes
  - 3.5. Dérivation de nouveaux types complexes
    - a) Dérivation par extension
    - b) Dérivation par restriction
  - 3.6. Groupes d'éléments et d'attributs
  - 3.7. Clés et références de clés

## Clés et références de clés

- Les contraintes de clé est une extension du ID/IDREF des DTD.
- Les schémas offrent la possibilité de définir avec une grande précision des clefs (avec **xs:unique** ou **xs:key**) et des références (avec **xs:keyref**)
- Se rapproche de la notion de clé **dans le relationnel**:
  - Contrainte d'unicité UNIQUE, de clef primaire (unicité et existence) PRIMARY KEY, de clef étrangère FOREIGN KEY ... REFERENCES ....
  - Une clef primaire est définie pour une relation, et elle composée d'un ou plusieurs attributs.
  - Une clef étrangère fait référence à des attributs d'une relation précise.

## Exemple

Document XML

```
<service>  
    <employe nom="dupont jean" id="E1"/>  
    <employe nom="dupont louis" id="E2" chef="E1"/>  
</service>
```

- Un employé travaille dans un ou plusieurs services
- Un employé est chef d'un service
- Un service a un seul chef qui est employé dans l'entreprise

Schéma XML  
Déclaration des éléments  
Déclaration des attributs  
Types de base  
Types simples  
Types complexes  
Dérivation de types complexes  
Groupes  
Clés et références

## Exemple

Document XML

```
<service>  
  <employe nom="dupont jean" id="E1"/>  
  <employe nom="dupont louis" id="E2" chef="E1"/>  
</service>
```

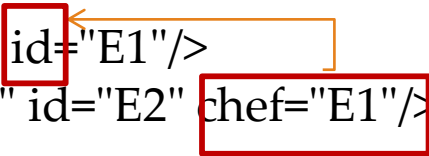


Schéma XML

```
<xs:element name="employee">  
  <xs:complexType>  
    <xs:attribute name="nom" type="xs:string"/>  
    <xs:attribute name="id" use="required" type="xs:string"/>  
    <xs:attribute name="chef" type="xs:string"/>  
  </xs:complexType>  
</xs:element>  
  
<xs:element name="service">  
  <xs:complexType>  
    <xs:sequence>  
      <xs:element ref="employee" maxOccurs="unbounded"/>  
    </xs:sequence>  
  </xs:complexType>  
</xs:element>
```

## Exemple

- Déclaration de la clé de l'employé

Schéma XML

```
<xs:element name="service">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="employe" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <!-- identifiant de l'employe -->
  <xs:key name="kid">
    <xs:selector xpath="employe"/>
    <xs:field xpath="@id"/>
  </xs:key>
</xs:element>
```

indique sur quels éléments  
est vérifiée la contrainte de clé

détermine où se situe la valeur qui  
doit être unique

**Les clés sont associées aux éléments (conteneurs) et non pas aux types !**

```
<xs:element name="service">  
  <xs:complexType>  
    <xs:sequence>  
      <xs:element ref="employe" maxOccurs="unbounded"/>  
    </xs:sequence>  
  </xs:complexType>  
  
  <!-- référence à l'identifiant de l'employé -->  
  <xs:keyref name="chef" refer="kid">  
    <xs:selector xpath="employe"/>  
    <xs:field xpath="@chef"/>  
  </xs:keyref>  
</xs:element>
```

**Les contraintes sont définies au niveau du conteneur des instances référencées !**

## Clés et références

Schéma XML complet

```
<xs:element name="service">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="employe" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
```

**<!-- identifiant de l'employe -->**

```
<xs:key name="kid">
  <xs:selector xpath="employe"/>
  <xs:field xpath="@id"/>
</xs:key>
```

détermine où se situe la valeur qui  
doit être unique

**<!-- référence à l'identifiant de l'employé -->**

```
<xs:keyref name="chef" refer="kid">
  <xs:selector xpath="employe"/>
  <xs:field xpath="@chef"/>
</xs:keyref>
</xs:element>
```

```

<xs:element name="service">
  <xs:complexType>
    ...
  </xs:complexType>
  <xs:key name="kid">
    <xs:selector xpath="employe"/>
    <xs:field xpath="@id"/>
  </xs:key>
</xs:element>

<xs:element name="entreprise">
  <xs:complexType>
    <xs:sequence maxOccurs="unbounded">
      <xs:element ref="service"/>
    </xs:sequence>
  </xs:complexType>
  <xs:keyref name="chef" refer="kid">
    <xs:selector xpath="service/employe"/>
    <xs:field xpath="@chef"></xs:field>
  </xs:keyref>
</xs:element>
  
```

Dans l'exemple suivant,  
 les définitions de la clé et  
 de la référence de clé sont  
 présentes à des niveaux  
 différents :

L'élément entreprise peut  
 contenir une référence de  
 clé vers l'attribut id de  
 l'élément  
 employe, car l'élément  
 entreprise est ancêtre de  
 l'élément employe.



## Clés et références de clés

- Les schémas offrent la possibilité de définir avec une grande précision des clefs (avec **xs:unique** ou **xs:key**) et des références (avec **xs:keyref**)
- `<xs:key>` Spécifie qu'un attribut ou qu'une valeur (ou ensemble de valeurs) d'élément doit être une clé dans la portée spécifiée.
- `<xs:unique>` Spécifie qu'une valeur (ou une combinaison de valeurs) d'attribut ou d'élément doit être unique dans la portée spécifiée. La valeur doit être unique ou null.
- L'élément `xs:key` est identique à `xs:unique` mais il impose en plus la présence obligatoire de la clef.
- Utilise (un sous ensemble) des expressions XPATH

## Clés et références

- `<xs:key>` comporte:
  - Un élément **<selector>** contenant une expression XPath afin d'indiquer l'élément à référencer.
  - Un ou plusieurs éléments `<xs:field>` contenant une expression XPath afin d'indiquer l'attribut servant d'identifiant.

```
<xs:key name= "nom_identifiant">  
<xs:selector xpath= "expression_XPath"></xs:selector>  
<!-- liste d'éléments field -->  
<xs:field xpath= "expression_XPath"></xs:field>  
</xs:key>
```

## Clés et références

- `<xs:keyref>` comporte:
  - Un élément `<xs:selector>` contenant une expression XPath afin d'indiquer l'élément à référencer.
  - Un ou plusieurs éléments `<xs:field>` contenant une expression XPath afin d'indiquer l'attribut servant d'identifiant.

```
<xs:keyref name= "nom" refer= "nom_identifiant">  
  <xs:selector xpath="expression_XPath"></xs:selector>  
  <!-- liste d'éléments field -->  
  <xs:field xpath= "expression_XPath"></xs:field>  
</xs:keyref>
```

- Keyref doit référencer une clé déclarée dans **le même élément ou dans un de ses descendants.**

# DTD VS XML SCHEMA

- DTD
  - Essentiellement, définition de l'imbrication des éléments et définition des attributs
  - Types pauvres
  - Pas de gestion des espaces de nom
  - Pas beaucoup de contraintes sur le contenu d'un document
- XML Schema
  - Notion de type, indépendamment de la notion d'élément
  - Contraintes de cardinalité
  - Réutilisation de mêmes types d'attributs pour des éléments différents
  - Format XML

# Liens utiles

- XML Schema Recommendation W3C: <http://www.w3.org/XML/Schema>
- Espaces de noms: <http://www.yoyodesign.org/doc/w3c/xml-namespace/Overview.html>
- Liste détaillée des types prédéfinis de XML Schema:  
<http://www.w3.org/TR/xmlschema-0/#CreatDt>  
[https://msdn.microsoft.com/fr-fr/library/ms256052\(VS.80\).aspx](https://msdn.microsoft.com/fr-fr/library/ms256052(VS.80).aspx)
- Support de Cours en ligne Elisabeth Murisasco
- Jacques Le Maitre, Description et manipulation de documents XML, supports de cours en ligne <http://lemaitre.univ-tln.fr/cours.htm>
- François Role « Modélisation et manipulation de documents XML » Lavoisier