Introduction à XML [DVIR3]

Informatique et systèmes finalité Réseaux & télécommunication

2016-2017

XML

1. Introduction
2. Langage XML
3. Document XML
4. DTD : Déclaration de la structure du document
5. XML Schema
6. Adressage de fragments xml : Xpath
7. Contenu et présentation : CSS et XSL
8. Langage de transformation XSLT
9. Langage de formatage XSL-FO
10. PHP - XML

XML

1. Introduction
2. Langage XML
3. Document xml
4. DTD : Déclaration de la structure du document
5. **XML Schema**
6. Adressage de fragments xml : Xpath
7. Contenu et présentation : CSS et XSL
8. Langage de transformation XSLT
9. Langage de formatage XSL-FO
10. PHP - XML

XML

1. Introduction
2. Langage XML
3. Document xml
4. DTD : Déclaration de la structure du document
5. **XML Schema**
6. Adressage de fragments xml : Xpath
7. Contenu et présentation : CSS et XSL
8. Langage de transformation XSLT
9. Langage de formatage XSL-FO
10. PHP - XML

Schema XML (ou Type)

Un schéma XML est un document XML
qui décrit d'autres documents XML !

Objectifs du Schema XML

Permet de définir des types de données

Il est donc plus simple de :

- décrire le contenu autorisé d'un document
- valider l'exactitude des données
- travailler avec des données issues d'une BD
- définir les restrictions aux données
- définir des modèles de données (data patterns)
- convertir des données entre différents types de données

XML DTD - Exemple

DTD : officiel.dtd

```
<!ELEMENT officiel (#PCDATA | cinema | film)*>
<!ELEMENT cinema (nom, adresse, (seance)*)>
...
```

XML Schema - **officiel.xsd**


```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd='http://www.w3.org/2001/XMLSchema'>
    <xsd:annotation>
        <xsd:documentation> Schéma pour officiel.com </xsd:documentation>
    </xsd:annotation>
    <xsd:element name='officiel' type='OfficielType'/>
    <xsd:complexType name='OfficielType' mixed='true'>
        <xsd:choice minOccurs='0' maxOccurs='unbounded'>
            <xsd:element name='cinema' type='CinemaType'/>
            <xsd:element name='film' type='FilmType'/>
        </xsd:choice>
     </xsd:complexType>
```

- officiel.xsd (2)


```
    <xsd:complexType name='CinemaType'>
        <xsd:sequence>
            <xsd:element name='nom' type='xsd:string'/>
            <xsd:element name='adresse' type='AdresseType'/>
            <xsd:element name='seance' type='SeanceType'
                        minOccurs='0' maxOccurs='unbounded'/>
        </xsd:sequence>
    </xsd:complexType>
....
</xsd:schema>
```

Schema XML *est* 1 document XML

- Séparation entre balises et types : différents éléments peuvent partager le même type
- (modèles de contenu)
- 43 Types simples, listes et union de types simple
- Constructeurs de types complexes: sequence, choice, group
- Restriction de types par motifs (wildcards)
- Classes d'équivalences entre éléments
- Mécanisme d'extension et raffinement de types
- Mécanismes de documentation

Types simples

43 types atomiques >< DTD: un seul type d'élément (#PCDATA)
et 10 types d'attributs

. xsd:string, xsd:byte, xsd:unsignedByte

. xsd:integer, xsd:long, xsd:float, xsd:double, ...

. xsd:boolean

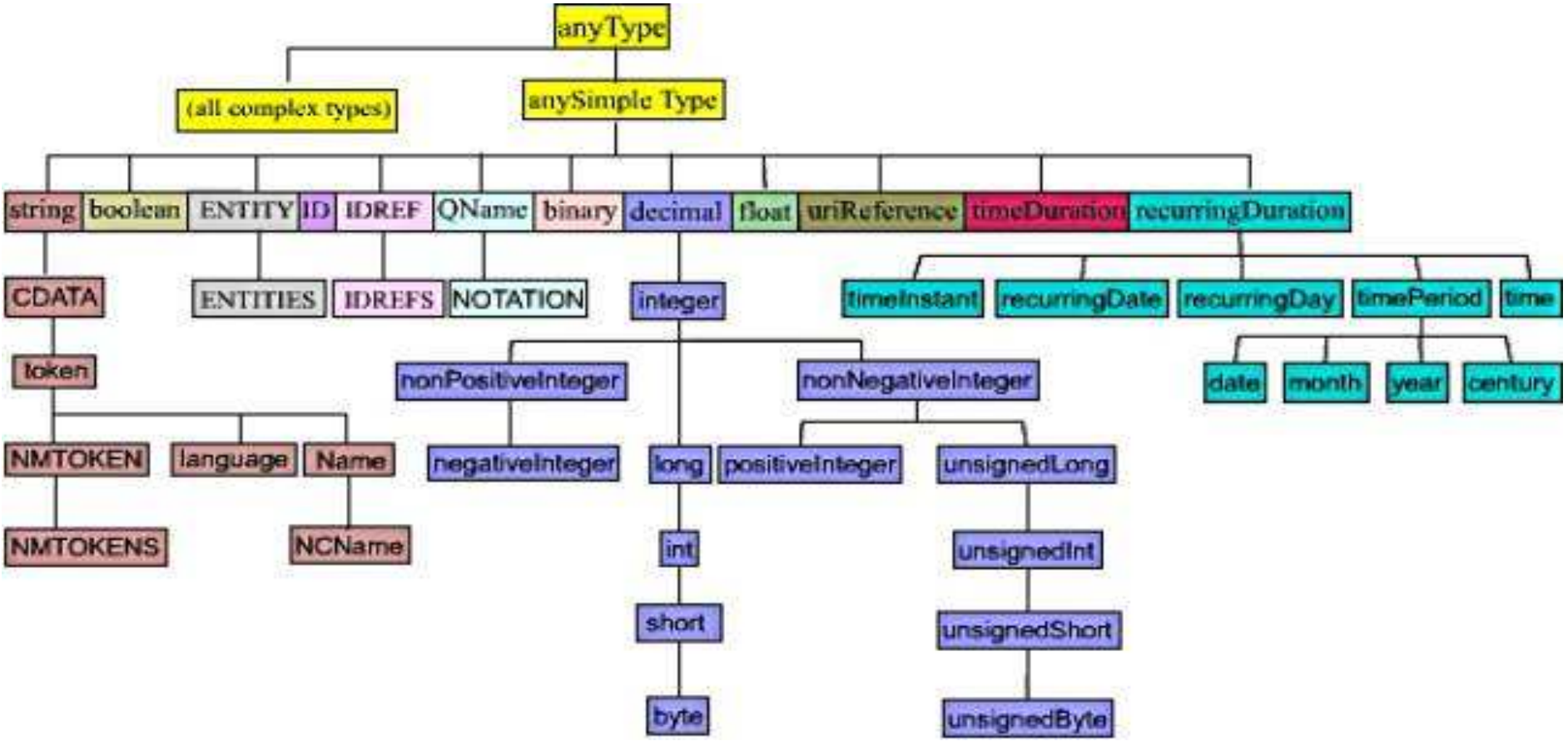
. xsd:time, xsd:timeDuration, xsd:Date, xsd:year, xsd:month, ...

. xsd:language, xsd:uriReference

. Types DTD: xsd:ID, xsd:IDREF, xsd:NMTOKEN, ...

. listes et unions de types simples

Types simples



Source :http://gilles.chagnon.free.fr/cours/xml/schema.html

Définition de types simples

- *Année Film: 1956*

- DTD: #PCDATA

- **XML schéma :** Restriction de type

```
<xsd:simpleType name='AnneeFilm'>
        <xsd:restriction base='xsd:integer'>
            <xsd:minInclusive value='1900'/>
            <xsd:maxInclusive value='2016'/>
        </xsd:restriction>
</xsd:simpleType>
```

Restrictions de types simples

- Longueur (length, minLength, maxLength) des chaînes de caractères ou listes
- Par *motifs* (chaînes de caractères)
- Enumération,
- Intervalles (maxInclusive, maxExclusive, minInclusive, minExclusive) et

- Autres (precision, scale, encoding, period, duration).

Restrictions par motif

- *numero de téléphone : +33-(0)-1-34-45-67-89*

- DTD : #PCDATA

- Schéma XML : Similaire aux **expressions régulières** de Perl, PHP, Python,JS,..

```
<xsd:simpleType name='NumTel'>
    <xsd:restriction base='xsd:string'>
        <xsd:pattern value='\+33-\(0\)-\d(-\d{2}){4}'/>
    </xsd:restriction>
</xsd:simpleType>
```

Restrictions par liste de valeurs

- *Element xml :<Annuaire>33-(0)-1-34-45-67-89*
  33-(0)-3-25-56-98-78</Annuaire>

- DTD: <!ELEMENT Annuaire #PCDATA> (attribut: NMTOKENS)

- XML Schéma:

```
<xsd:simpleType name='ListeNumTel'>
    <xsd:list itemType='NumTel'/>
</xsd:simpleType>
```

Restriction de Listes par Longueur

- *Listes avec 5 numeros de téléphone:*

- XML Schéma:

```
<xsd:simpleType name='CinqNumTel'>
    <xsd:restriction base='ListNumTel'>
        <xsd:length value='5'/>
    </xsd:restriction>
</xsd:simpleType>
```

Union de Types Simples

- *numeros européens:* *+33-(0)-1-45-67-89*
  +49-(0)-55-50-24-91
  +32-(0)2-219-15-46

- DTD: #PCDATA ou NMTOKENS (attributs)

- XML schéma:

```
<xsd:simpleType name='EuroNumTel'>
    <xsd:union memberTypes='FrancTel GerTel BelTel...'/>
</xsd:simpleType>
```

Types Complexes, Éléments, Attributs

- Constructeur de type: xsd:sequence, xsd:choice, xsd:group
- Déclaration d'éléments:

  <xsd:element name type contraintes [value]/>

  <xsd:element ref contraintes/>

  Contraintes: minOccurs, maxOccurs

  Value: fixed, default
- Attributs:

  <xsd:attribut name use [value]/>

  Use: required, optional, prohibited

  Value: fixed, default

Type Complexe: Exemple

- Modèle de contenu
- DTD: (titre, annee)
- XML Schéma :

```
<xsd:complexType name='FilmType'>
    <xsd:sequence>
        <xsd:element name='titre' type='xsd:string'/>
        <xsd:element name='annee'>
            <xsd:simpleType>
                <xsd:restriction base='xsd:year'>
                    <xsd:maxInclusive value='2016'/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
```

Types simples avec attributs

- Élément XML:

  <titre langue='français'>Le Goût des Autres</titre>

- DTD:

  <!ELEMENT titre #PCDATA>

  <!ATTLIST titre langue NMTOKEN>

Types simples avec attributs

- Élément XML:

```
<titre langue='français'>Le Goût des Autres</titre>
```

- **XML schéma**: *les types simples (e.g. xsd:string) n'ont pas d'attributs :*

```
<xsd:element name='titre'>
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base='xsd:string'>
                <xsd:attribute name='langue' type='xsd:string' />
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
```

Contenu mélangé

- **Élément XML**:

  <Officiel> cinemas: <cinema> ... </cinema> Films: <film>... </film> </officiel>

- **DTD**: <!ELEMENT Officiel (#PCDATA | cinema | film)*>

- **Schéma XML** (<xsd:choice> peut être remplacé par xsd:sequence) :

```
<xsd:complexType name='OfficielType' mixed='true'>
    <xsd:choice minOccurs='0' maxOccurs='unbounded'>
        <xsd:element name='cinema' type='CinemaType' />
        <xsd:element name='film' type='FilmType' />
    </xsd:choice>
</xsd:complexType>
```

Contenu vide

- *Élément XML: <film titre='If' année='1976'/>*
- Schéma XML:

```
<xsd:element name='film'>
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:restriction base='xsd:anyType'>
                <xsd:attribute name='titre' type='xsd:string'/>
                <xsd:attribute name='annee' type='AnneeFilm'/>
            <xsd:restriction>
        </xsd:complexContent>
    </xsd:complexType>
</xsd:element>
```

Groupes: Exemple

- DTD: <!ELEMENT A ((B|C)*,D)+>
- XML schéma:

```
<xsd:element name='A'>
    <xsd:complexType>
        <xsd:group minOccurs='1' maxOccurs='unbounded'>
            <xsd:sequence>
                <xsd:group minOccurs='0' maxOccurs='unbounded'>
                    <xsd:choice>
                        <xsd:element name='B' xsd:type='xsd:string'/>
                        <xsd:element name='C' xsd:type='xsd:string'/>
                    </xsd:choice>
                </xsd:group>
                <xsd:element name='D' xsd:type='xsd:string'/>
            </xsd:sequence>
        </xsd:group>
    </xsd:complexType>
</xsd:element>
```

Groupe d'attributs

- L'adresse d'un cinema ou d'une personne est composée des mêmes attributs (partage)

```
<xsd:element name='cinema'>
        ...     <xsd:attributeGroup ref='Adresse'/>
</xsd:element>
<xsd:element name='personne'>
        ...     <xsd:attributeGroup ref='Adresse'/>
</xsd:element>
<xsd:attributeGroup name='Adresse'>
        <xsd:attribute name='ville' type='xsd:string'/>
        <xsd:attribute name='rue' type='xsd:string'/>
        <xsd:attribute name='numero' type='xsd:decimal'/>
</xsd:attribute>
```

Valeurs Nulles

- **Schéma XML:**

```
<xsd:element name='heureFilm'
    type='xsd:time' nillable='true'/>
```

- **Élément XML**:

```
<heureFilm xsi:nil='true'/>
```

xsi:null est défini dans le domaine de noms pour des instances http://www.w3.org/2001/XMLSchema-instance

Valeur unique

- L'identificateur d'un film doit être unique:

```
<unique name='toto'>
    <selector>film</selector>
    <field>film_id</field>
</unique>
```

- La valeur de <field> doit être unique à l'intérieur de chaque élément sélectionné par le sélecteur.
- Les valeurs des element <selector> et <field> sont des expressions XPath.

Clés et références

- Une clé est unique (filmcle attribut clé d'une relation):

```
<key name='filmcle'>
    <selector>film</selector>
    <field>@film_id</field>
</key>
```

- Référence (filmref attribut clé étrangère d'une relation)

```
<keyref name='filmref' refer='filmcle'>
    <selector>seance</selector>
    <field>@ref_film</field>
</keyref>
```

Exemple


```
<book identifier="isbn-0836217462">
    <isbn>0836217462</isbn>
    <title>Being a Dog Is a Full-Time Job</title>
    <author-ref ref="au-Charles_M._Schulz"/>
    <character-refs>ch-Peppermint_Patty ch-Snoopy ch-Schroeder ch-Lucy
    </character-refs>
</book>
```

Src : http://examples.oreilly.com/xmlschema/

Exemple


```
<xs:element name="book">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="isbn" type="xs:int"/>
            <xs:element name="title" type="xs:string"/>
            <xs:element name="author-ref">
                <xs:complexType>
                    <xs:attribute name="ref" type="xs:IDREF“/>
                </xs:complexType>
            </xs:element>
            <xs:element name="character-refs" type="xs:IDREFS"/>
        </xs:sequence>
        <xs:attribute name="identifier" type="xs:ID" use="required"/>
    </xs:complexType>
</xs:element>
```

Exemple : **Keys and Key References**


```
<xs:element name="book" maxOccurs="unbounded">
    <xs:complexType>
        .../...
    </xs:complexType>

    <xs:unique name="book">
        <xs:selector xpath="book"/>
        <xs:field xpath="isbn"/>
    </xs:unique>
</xs:element>
```

Exercice

Définir la validité d'un document .xml mettant en évidence la structure des cours à l'ESI à l'aide d'un schéma :

1. Utiliser des attributs de type ID et IDREF(S)
2. Modifier le document et le schéma pour assurer les contraintes référentielles en n'utilisant que key et keyref

Domaines Nominaux


```
<schema xmlns='http://www.w3.org/1999/XMLSchema'
    targetNamespace='http://www.officiel.com/namespace'
    xmlns:officiel='http://www.officiel.com/namespace'>
    <element name='cinema'>
        <complexType type='CinemaType' content='elementOnly'>
            <element name='nom' type='string'/>
            <element ref='officiel:adresse'/>
            <sequence minOccurs='0' maxOccurs='unbounded'>
                <element ref='officiel:seance'/>
            </sequence>
        </complexType>
    </element>
</schema>
```

Éléments locaux /globaux

- **Éléments locaux** : définis au travers d'un type complexe, qualifiés ou non par un préfixe de *namespace* en fonction de l'attribut *elementFormDefault* (*qualified* ou *unqualified*)

- **Éléments globaux** : définis directement au 1er niveau (sous schema) et toujours qualifiés avec un préfixe de namespace

Éléments Locaux Qualifiés et Non-qualifiés

Document XML:

```
<?xml version='1.0'?>
<officiel:officiel xmlns:officiel='http://www.officiel.com/namespace'>
    <officiel:cinema>
        <nom>St André des Arts</nom>
        <officiel:adresse>
            <ville> Paris </ville>
            <rue> rue St. André des Arts </rue>
            <numero> 13 </numero>
        </officiel:adresse>
    </officiel:cinema>
</officiel:officiel>
```

Élément non qualifié

- Les éléments *nom, ville, rue, numero* ne sont pas qualifiés
- La clause *elementFormDefault='qualified'* dans le schéma permet de forcer la qualification des éléments locaux et donc, comme dans une DTD, il n'est plus possible d'avoir deux éléments avec le même nom

Extension de Types Complexes

- cinemas avec un site Web et un pays

```
<include schemaLocation='http://www.officiel.com/officiel.xsd' />
<complexType name='CyberCinemaType'>
    <complexContent>
        <extension base='officiel:CinemaType'>
            <sequence>
                <element name='webaddress' type='url'/>
                <element name='country' type='string'/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
```

La clause <include>

- Permet d'importer un schéma avec le même domaine nominal cible (target namespaces)

- Le type <CyberCinemaType> est un sous-type du type <CinemaType>

Utilisation de types étendus

- Le type d'un élément est défini explicitement avec l'attribut xsi:type

```
<officiel:cinema xsi:type='CyberCinemaType'>
    <nom>St André des Arts</nom>
    <officiel:adresse>
        <ville> Paris </ville>
        <rue> rue St. André des Arts </rue>
        <numero> 13 </numero>
    </officiel:adresse>
    <url>http://www.st-andré.com</url>
</officiel:cinema>
```

Restriction de Types Complexes

- Cinemas avec une à trois seances

```
<include schemaLocation='http://www.officiel.com/officiel.xsd'/>
<complexType name='SeanceobligType'>
    <complexContent>
        <restriction base='officiel:CinemaType'>
            <complexType content='elementOnly'>
                <sequence minOccurs='1' maxOccurs='3'>
                    <element ref='officiel:seance'/>
                </sequence>
                <element name='nom' type='string'/>
                <element ref='officiel:adresse'/>
            </complexType>
        </restriction >
    </complexContent>
</complexType>
```

Redéfinitions de types

- cinemas avec une url (chaîne de caractères) :

```
<redefine schemaLocation='http://www.officiel.com/officiel.xsd' >
    <complexType name='CinemaType'>
        <complexContent>
            <extension base='officiel:CinemaType'>
                <sequence>
                    <element name='webaddress' type='url'/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</redefine>
```

- Attention: il est possible de créer des conflits avec des types dérivés du type redéfini: deux éléments webaddress avec le type CyberCinemaType

Groupes de substitution

Substitution d'éléments par d'autres éléments:

- XML schéma:

```
<element name='movieTheatre'     type='officiel:CinemaType'
                                 substitutionGroup='officiel:cinema'/>
```

- L'élément *movieTheatre* peut remplacer *cinema*.

Types et Groupes Abstraits

- Schéma XML: il est possible de définir des éléments abstraits qui ne peuvent pas être instantiés directement :

```
<complexType name='CulturalplaceType' abstract='true'/>
<complexType name='CinemaType'>
    <complexContent>
        <extension base='CulturalplaceType'/>
        ...
    </complexContent>
</complexType>
<element name='culturalplace' type='officiel:CulturalplaceType'/>
```

- Dans le document XML, le type doit être spécifié par l'attribut xsi:type :

```
<culturalplace xsi:type='CinemaType'/>
```

Espace de noms ou Domaines nominaux

- Un domaine nominal XML (namespace) est une collection de noms d'éléments ou noms d'attributs (identifiée par un URI)
- Utilisation : éviter les conflits de noms
- Déclaration du namespace : au début du document l'utilisant

exemple

Définition de l'espace de noms utilisé :

<document xmlns:ATDB= 'http://www.brol.ac.be/atdb-schema'>

Utilisation d'une balise de l'espace de noms :

<ATDB: monTAG>Ceci est un tag xml.</ ATDB: monTAG>

Utilisation d'une balise hors du namespace :

<monTAG>Ceci est un tag fait maison.</monTAG>

Exemple


```
<schema targetNamespace="http://www.automobile.fr/namespace/cmmd"
        xmlns="http://www.automobile.org/xmlschema-commande-2000-001"
        xmlns:cm="http://www.automobile.fr/namespace/cmmd">
    <element name="Commande" type="cm:Commandetype"/>
    <element name="Commentaire" type="string"/>
    <type name="Commandetype" >
        <element name="Client" type="cm:Adresse" />
        <element name="Datelivraison" type="date" />
        <element ref="cm:Commentaire" minOccurs="0" />
        <element name="Items" type="cm:Items" />
        <attribute name="DateCommande" type="date" />
    </type>
```

Exemple - suite


```
<type name="Adresse" >
    <element name="Nom" type="string" />
    <element name="Rue" type="string" />
    <element name="Codepostal" type="integer" />
    <element name="Ville" type="string" />
    <element name="Pays" type="string" />
    <attribute name="type" type="string" />
</type>
```

Exemple - suite


```
<type name="Items" >
    <element name="Item" minOccurs="0" maxOccurs="*" >
        <type>
            <element name="NomProduit" type="string" />
            <element name="Quantite" >
                <datatype source="integer">
                    <minExclusive value="0" />
                </datatype>
            </element>
            <element name="Prix" type="decimal " />
            <element ref="po:Commentaire" minOccurs="0" />
            <attribute name="DateCommande" type="date" />
        </type>
    </element>
</type>

</schema>
```

Déclaration de schéma


```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.w3schools.com"
   xmlns="http://www.w3schools.com"
   elementFormDefault="qualified">
 ...
</xs:schema>
```

note.xml avec *note.xsd*


```
<?xml version="1.0"?>
<note xmlns="http://www.w3schools.com"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3schools.com note.xsd">
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
</note>
```

note.xml avec ***note.xsd***


```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           targetNamespace="http://www.w3schools.com"
           xmlns="http://www.w3schools.com" elementFormDefault="qualified">
    <xs:element name="note">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="to" type="xs:string"/>
                <xs:element name="from" type="xs:string"/>
                <xs:element name="heading" type="xs:string"/>
                <xs:element name="body" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```