

Le Standard XML -DTD-

Ichrak MEHREZ
(m_ichrak@hotmail.fr)

Rappel

- Le XML nous permet de créer notre propre vocabulaire grâce à un ensemble de règles et de balises personnalisables.
- Document XML valide: un document XML qui est *conforme aux règles syntaxiques*
- Les **fichiers de définition**: définir une structure stricte aux documents XML : les **DTD** et les **schémas XML**.

Rôles du DTD

- **DTD** (*D*ocument *T*ype *D*efinition) : permet de définir précisément la structure d'un document.
- Il s'agit d'un certain nombre de contraintes que doit respecter un document pour être *valide*.
- Ces contraintes spécifient quels sont les éléments qui peuvent apparaître dans le contenu d'un élément, l'ordre éventuel de ces éléments et la présence de texte brut.
- Elles définissent aussi, pour chaque élément, les attributs autorisés et les attributs obligatoires.

DTD vs Schémas XML

- Les DTD ont l'avantage d'être relativement simples à utiliser mais elles sont parfois aussi un peu limitées.
- *Les schémas XML* permettent de décrire de façon plus précise encore la structure d'un document. Ils sont plus sophistiqués mais plus difficiles à mettre en œuvre.
- Les DTD sont donc particulièrement adaptées pour des petits modèles de documents.

Syntaxe générale

- La DTD contient des déclarations d'éléments et d'attributs délimitées par les chaînes de caractères '<!' et '>'.
■ Un mot clé juste après la chaîne '<!' indique le type de la déclaration.

```
<?xml version="1.0"?><!DOCTYPE books [  
  <!ELEMENT title (#PCDATA)>  
  <!ELEMENT author (#PCDATA)>  
  <!ELEMENT authors (author)+>  
  <!ELEMENT subject (#PCDATA)>  
  <!ATTLIST subject class CDATA "">  
  <!ELEMENT book (title,authors,subject)>  
  <!ATTLIST book  
    bookid CDATA #REQUIRED  
    pubdate CDATA #REQUIRED>  
<books name="My books">  
  <book bookid="1" pubdate="03/01/2002">  
    <title>Java Web Services</title>  
    <authors>
```

```
<?xml version="1.0"?>  
<!DOCTYPE person [  
  <!ELEMENT person (age,gender,name)>  
  <!ELEMENT age (#CDATA)>  
  <!ELEMENT gender (#PCDATA)>  
  <!ELEMENT name (#PCDATA)>  
  
<person>  
  <age>40</age>  
  <gender>male</gender>  
  <name>Peter Miller</name>  
</person>
```

Déclaration de la DTD

- La déclaration de la DTD du document doit être placée dans *le prologue*.
- La DTD peut être interne, externe ou mixte:
 - *interne* si elle est directement incluse dans le document.
 - *externe* si le document contient seulement une référence vers un autre document contenant la DTD.
 - *mixte* si elle est constituée d'une partie interne et d'une partie externe.

Déclaration de la DTD

- La déclaration de la DTD est introduite par le mot clé **DOCTYPE** et a la forme générale suivante où *root-element* est le nom de l'élément racine du document.

<!DOCTYPE root-element ... >

- Le nom de l'élément racine est suivi du contenu de la DTD dans le cas d'une DTD interne ou de l'URL du fichier contenant la DTD dans le cas d'une DTD externe.

DTD interne

- Lorsque la DTD est incluse dans le document, sa déclaration prend la forme suivante où son contenu est encadré par des caractères crochets '[' et ']'.
[et]

```
<!DOCTYPE root-element [ declarations ]>
```

- Exemple : le nom de l'élément racine est *personne*. La DTD déclare que cet élément ne peut contenir que du texte (***Parsed Characters DATA***) et pas d'autre élément.

```
<!DOCTYPE personne [  
    <!ELEMENT personne (#PCDATA)>  
>
```


DTD externe

- Lorsque la DTD est externe, celle-ci est contenue dans un autre fichier dont l'extension est généralement **.dtd**.
- NB: la première ligne doit faire apparaître l'attribut *standalone* avec la valeur *no*.
- Deux types:
 - **SYSTEM** :le fichier spécifié se trouve sur l'ordinateur local et qu'il est disponible uniquement à titre privé.

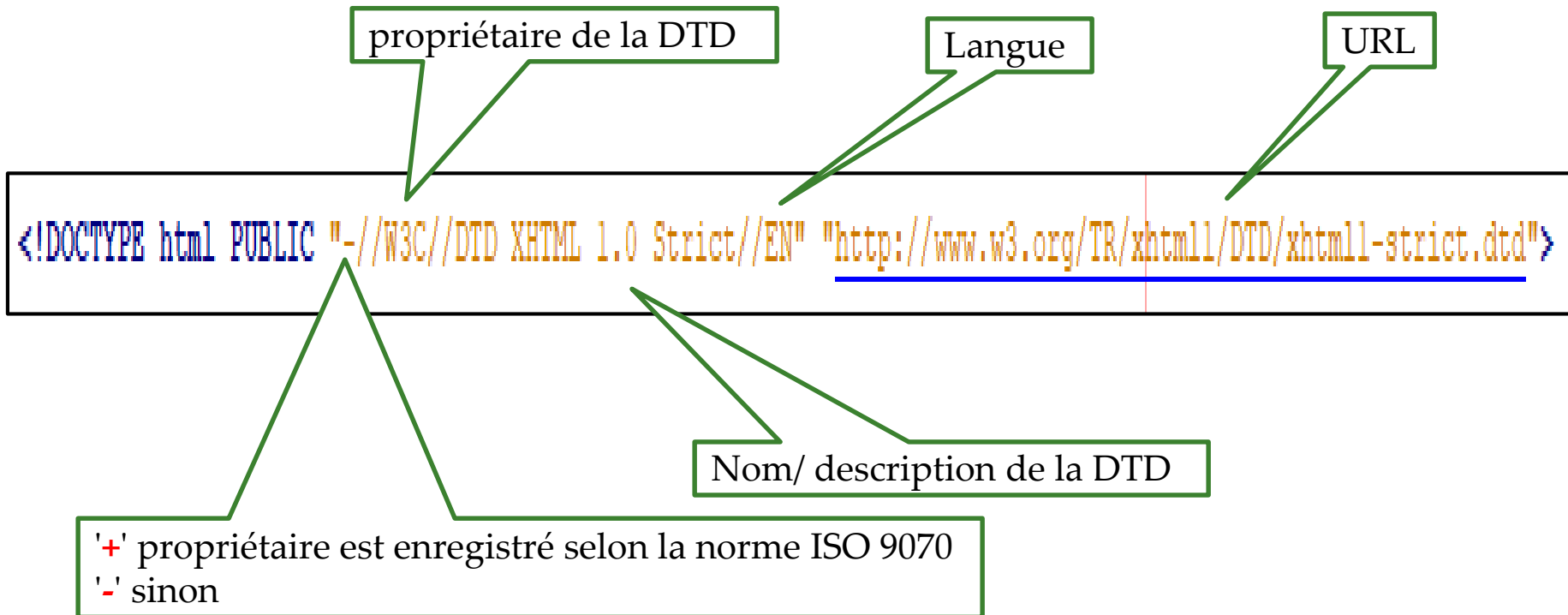
```
<!DOCTYPE personne SYSTEM "personne.dtd">
```

- **PUBLIC** :une ressource disponible pour tous sur un serveur distant.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

DTD externe

DTD PUBLIC:



DTD mixte

- Lorsque la DTD est mixte, tout *élément* doit être déclaré dans la partie interne ou dans la partie externe mais pas dans les deux.
- En revanche, il est possible de déclarer plusieurs fois le même *attribut*. La première déclaration a priorité.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE personne SYSTEM "personne.dtd" [
    .
    .
    .
] >

<personne>
    .
    .
    .
</personne>
```

Contenu de la DTD

- Une DTD est constituée de déclarations d'*éléments*, d'*attributs* et d'*entités*.
- Chacune de ces déclarations commence par la chaîne '<!' suivi d'un *mot clé* qui indique le type de déclaration.
- La déclaration se termine par le caractère '>'.
- Les mots clés possibles sont **ELEMENT**, **ATTLIST** et **ENTITY**.

Déclaration d'éléments (1/7)

- Il s'agit ici de déclarer les éléments autorisés à apparaître dans le document, ainsi que leurs imbrications possibles.
- La déclaration d'un élément prend la forme suivante

```
<!ELEMENT element type>
```
- *element* : nom de l'élément, doit être un nom XML
- *type* : détermine les *contenus valides* de l'élément:
 - ❑ *contenus purs* uniquement constitués d'autres éléments,
 - ❑ *contenus textuels* uniquement constitués de texte,
 - ❑ *contenus mixtes* qui mélangent éléments et texte.

Déclaration d'éléments (2/7)

Contenus purs

- Le contenu d'un élément est *pur* lorsqu'il ne contient aucun texte.
- Ces enfants peuvent, à leur tour, avoir des contenus *textuels*, *purs* ou *mixtes*,
- La déclaration d'un élément de contenu *pur* détermine quels peuvent être ses enfants et dans quel ordre ils doivent apparaître.
- Le contenu de chaque enfant est décrit par sa propre déclaration.

Déclaration d'éléments (3/7)

Contenus purs

- *element* : nom de l'élément
- *regex* : décrit les suites autorisées d'enfants dans le contenu de l'élément.
- *regex* est construite à partir des noms d'éléments en utilisant les opérateurs *,*, *|*, *?*, *** et *+* ainsi que les parenthèses *(* et *)* pour former des groupes.

`<!ELEMENT element regex>`

| Opérateur | Signification |
|-----------|---|
| <i>,</i> | Mise en séquence |
| <i> </i> | Choix |
| <i>?</i> | 0 ou 1 occurrence |
| <i>*</i> | Répétition d'un nombre quelconque d'occurrences |
| <i>+</i> | Répétition d'un nombre non nul d'occurrences |

Déclaration d'éléments (4/7)

Contenus purs

`<!ELEMENT elem (elem1, elem2, elem3)>`

L'élément elem doit contenir un élément elem1, un élément elem2 puis un élément elem3 dans cet ordre.

`<!ELEMENT elem (elem1 | elem2 | elem3)>`

L'élément elem doit contenir un seul des éléments elem1, elem2 ou elem3.

`<!ELEMENT elem (elem1, elem2?, elem3)>`

L'élément elem doit contenir un élément elem1, un ou zéro élément elem2 puis un élément elem3 dans cet ordre.

`<!ELEMENT elem (elem1, elem2*, elem3)>`

L'élément elem doit contenir un élément elem1, une suite éventuellement vide d'éléments elem2 et un élément elem3 dans cet ordre.

`<!ELEMENT elem (elem1, (elem2 | elem4), elem3)>`

L'élément elem doit contenir un élément elem1, un élément elem2 ou un élément elem4 puis un élément elem3 dans cet ordre.

`<!ELEMENT elem (elem1, elem2, elem3)*>`

L'élément elem doit contenir une suite d'éléments elem1, elem2, elem3, elem1, elem2, ... jusqu'à un élément elem3.

`<!ELEMENT elem (elem1 | elem2 | elem3)*>`

L'élément elem doit contenir une suite quelconque d'éléments elem1, elem2 ou elem3.

`<!ELEMENT elem (elem1 | elem2 | elem3)+>`

L'élément elem doit contenir une suite non vide d'éléments elem1, elem2 ou elem3.

Déclaration d'éléments (5/7)

Contenus textuels

- La déclaration de la forme suivante indique qu'un élément peut uniquement contenir du texte.

```
<!ELEMENT element (#PCDATA)>
```

- Ce texte est formé de *caractères*, d'*entités* qui seront remplacées au moment du traitement et de *sections littérales*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Personnes [
    <!ELEMENT Personnes (personne)*>
    <!ELEMENT personne (nom,prenom)>
    <!ELEMENT nom (#PCDATA)>
    <!ELEMENT prenom (#PCDATA)>
]>
<Personnes>
    <personne>
        <nom>Tounsi</nom>
        <prenom>Ali</prenom>
    </personne>
    <personne>
        <nom>Gharbi</nom>
        <prenom>Sami</prenom>
    </personne>
</Personnes>
```

Déclaration d'éléments (6/7)

Contenus mixtes

```
<!ELEMENT element (#PCDATA | element1 | ... | elementN)*>
```

- C'est la seule façon d'avoir un contenu mixte avec du texte et des éléments.
- Il n'y a aucun contrôle sur le nombre d'occurrences de chacun des éléments et sur leur ordre d'apparition dans le contenu de l'élément ainsi déclaré.
- Le mot clé **#PCDATA** doit apparaître en premier avant tous les noms des éléments.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Personnes [
    <!ELEMENT Personnes (personne)*>
    <!ELEMENT personne (#PCDATA|nom|prenom)*>
    <!ELEMENT nom (#PCDATA)>
    <!ELEMENT prenom (#PCDATA)>
]>
<Personnes>
  <personne>
    Monsieur <nom>Tounsi</nom>
               <prenom>Ali</prenom>
    est inscrit à l'Université de Tunis elManar
  </personne>
  <personne>
    Madame <nom>Gharbi</nom>
            <prenom>Sami</prenom>
    est inscrite à l'Ecole Polytechnique
    Internationale Privée de Tunis
  </personne>
</Personnes>
```

Déclaration d'éléments (7/7)

Contenus vides et contenus libres

- **Contenu vide**: la déclaration suivante indique que le contenu de l'élément *element* est nécessairement vide. Cet élément peut uniquement avoir des attributs. Les éléments vides sont souvent utilisés pour des liens entre éléments.

```
<!ELEMENT element EMPTY>
```

- **Contenu libre**: La déclaration suivante n'impose aucune contrainte sur le contenu de l'élément *element*. En revanche, ce contenu doit, bien entendu, être bien formé et les éléments contenus doivent également être déclarés.

```
<!ELEMENT element ANY>
```

Déclaration d'attributs (1/4)

- Tout attribut présent dans la balise ouvrante d'un élément doit être déclaré. La déclaration d'attribut prend la forme générale suivante:

```
<!ATTLIST element attribut type default>
```

- ❑ *attribut* est le nom de l'attribut,
 - ❑ *element* le nom de l'élément auquel il appartient,
 - ❑ *type* le type de l'attribut,
 - ❑ *default* la valeur par défaut de l'attribut.
- Il est possible de déclarer simultanément plusieurs attributs pour un même élément.

```
<!ATTLIST element attribut1 type1 default1  
                  attribut2 type2 default2  
                  ...  
                  attributN typeN defaultN>
```

Déclaration d'attributs (2/4)

Types des attributs

■ CDATA

- Ce type est le plus général. Il n'impose aucune contrainte à la valeur de l'attribut. Celle-ci peut être une chaîne quelconque de caractères.

■ (*value-1* | *value-2* | ... | *value-N*)

- La valeur de l'attribut doit être un des jetons *value-1*, *value-2*, ... *value-N*. Celles-ci ne sont pas délimitées par des apostrophes "'" ou des guillemets '"

Un *jeton* est une suite quelconque des caractères suivants: [a-z], [A-Z], [0-9] ainsi que le tiret '-', le point '.', les deux points ':' et le tiret souligné '_

■ NMTOKEN

- La valeur de l'attribut est un jeton

■ NMTOKENS

- La valeur de l'attribut est une liste de jetons séparés par des espaces.

Déclaration d'attributs (3/4)

Types des attributs

■ ID

- La valeur de l'attribut est un nom XML. Un élément peut avoir un seul attribut de ce type.

■ IDREF

- La valeur de l'attribut est une référence à un élément identifié par la valeur de son attribut de type ID.

■ IDREFS

- La valeur de l'attribut est une liste de références séparées par des espaces.

■ NOTATION

- La valeur de l'attribut est une notation

■ ENTITY

- La valeur de l'attribut une entité externe non XML

■ ENTITIES

- La valeur de l'attribut une liste d'entités externes non XML

Déclaration d'attributs (4/4)

Valeurs par défaut

- *"value"* ou *'value'*
 - La valeur *value* est une chaîne quelconque de caractères. Si l'attribut est absent pour un élément du document, sa valeur est implicitement la chaîne *value*. Cette valeur doit, bien sûr, être du type donné à l'attribut.
- **#IMPLIED**
 - L'attribut est *optionnel* et il n'a pas de valeur par défaut. Si l'attribut est absent, il n'a pas de valeur.
- **#REQUIRED**
 - L'attribut est *obligatoire* et il n'a pas de valeur par défaut.
- **#FIXED** *"value"* ou **#FIXED** *'value'*
 - La valeur de l'attribut est fixée à la valeur *value* donnée. Si l'attribut est absent, sa valeur est implicitement *value*. Si l'attribut est présent, sa valeur doit être *value* pour que le document soit valide.

Exemple

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Personnes[
    <!ELEMENT Personnes (personne)*>
    <!ELEMENT personne (nom,prenom,epoux?)>
        <!ATTLIST personne cin ID #REQUIRED
            sexe (M|F) #REQUIRED>

    <!ELEMENT nom (#PCDATA)>
        <!ATTLIST nom nomDeFamille CDATA #IMPLIED>
    <!ELEMENT prenom (#PCDATA)>
    <!ELEMENT epoux EMPTY>
        <!ATTLIST epoux idEpoux IDREF #REQUIRED>
    ] >
<Personnes>
    <personne cin="id-12345678" sexe='M'>
        <nom>Tounsi</nom>
        <prenom>Ali</prenom>
    </personne>
    <personne cin="id-08963254" sexe='F'>
        <nom nomDeFamille="Ben Younes">Gharbi</nom>
        <prenom>Samia</prenom>
        <epoux idEpoux="id-12345678"/>
    </personne>
</Personnes>
```


Les entités (1/5)

- Le principe sur lequel est fondé le système des **entités** est celui du **remplacement** d'une chaîne de caractères par un **symbole**, puis de l'utilisation du symbole à la place de cette chaîne.

```
<!ENTITY nom_entité "texte de remplacement">
```

- Une entité de nom *name* est référencée, c'est-à-dire utilisée, par **&name;** où le nom *name* de l'entité est encadré par les caractères '**&**' et '**;**'

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Etudiants[
    .
    <!ELEMENT universite (#PCDATA)>
    <!ENTITY pi "Ecole Polytechnique Internationale Privée de Tunis">
    ] >
<Etudiants>
    .
    <universite> &pi; </universite>
    .
</Etudiants>
```

Les entités (2/5)

Entités prédéfinies

- Il existe des entités prédéfinies permettant d'inclure les caractères spéciaux '<', '>', '&', "'" et '"' dans les contenus d'éléments et dans les valeurs d'attributs.

| Entité | Caractère |
|--------|-----------|
| < | < |
| > | > |
| & | & |
| ' | ' |
| " | " |


Les entités (3/5)

Entités internes

- La valeur d'une entité interne est le fragment de document associé à celle-ci lors de sa déclaration. Cette valeur peut contenir des caractères ainsi que des éléments avec leurs balises.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Universites[
  <!--
    <!--ELEMENT Universites (universite)*>
    <!--ENTITY pi '<name>Ecole Polytechnique Internationale
      Privée de Tunis</name>
      <adresse>Rue du Lac d Annecy Tunis 1053</adresse>'
    -->
    <!--ELEMENT universite ANY>
    <!--ELEMENT name (#PCDATA)>
    <!--ELEMENT adresse (#PCDATA)>
  ] >

  <Universites>
    <universite>
      &pi;
    </universite>
  </Universites>
```



```
▼ <Universites>
  ▼ <universite>
    <name>Ecole Polytechnique Internationale Privée de Tunis</name>
    <adresse>Rue du Lac d Annecy Tunis 1053</adresse>
  </universite>
</Universites>
```

Les entités (4/5)

Entités externes

- Une entité peut désigner un fragment de document contenu dans un autre fichier. Ce mécanisme permet de répartir un même document sur plusieurs fichiers comme dans l'exemple suivant.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE book [
  <!-- Entités externes -->
  <!ENTITY chapter1 SYSTEM "chapter1.xml">
  <!ENTITY chapter2 SYSTEM "chapter2.xml">
]>
<book>
  <!-- Inclusion du fichier chapter1.xml -->
  &chapter1;
  <!-- Inclusion du fichier chapter2.xml -->
  &chapter2;
</book>
```

Les entités (5/5)

Entités paramètres

- Les *entités paramètres* sont des entités qui peuvent uniquement être utilisées à l'intérieur de la DTD.
- La seule différence avec la déclaration d'une entité générale est la présence du caractère **'%'** entre le mot clé *ENTITY* et le *nom de l'entité* déclarée.

```
<!-- Déclaration de deux entités paramètres -->  
<!ENTITY % idatt "id ID #REQUIRED">  
<!ENTITY % langatt "xml:lang NMTOKEN 'fr'">  
<!-- Utilisation des deux entités paramètres -->  
<!ATTLIST chapter %idatt; %langatt;>
```