Bases de données et web – CM2

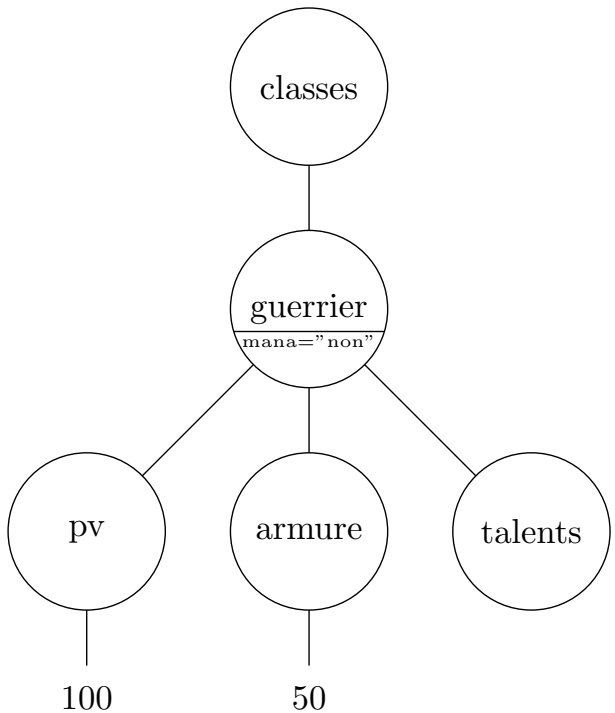
L3 Informatique & L3 MIASHS

DTD

Le langage XML permet de décrire un ensemble de données, mais n'attache pas directement de sens (sémantique) à son contenu.

Une première étape pour donner du sens à un jeu de données structuré est de spécifier les règles de la structure en question.

Par exemple, si on autorise tout, dans l'arbre ci-dessous, rien n'empêche l'élément `guerrier` de posséder deux enfants `pv`. Ce qui visiblement dans ce contexte, n'aurais aucun sens.

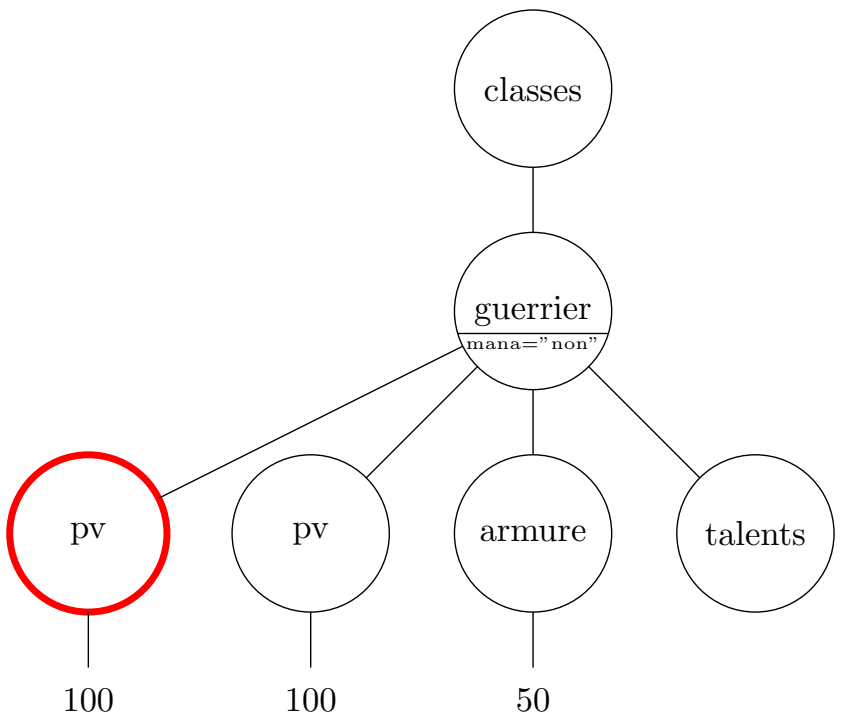


DTD

Le langage XML permet de décrire un ensemble de données, mais n'attache pas directement de sens (sémantique) à son contenu.

Une première étape pour donner du sens à un jeu de données structuré est de spécifier les règles de la structure en question.

Par exemple, si on autorise tout, dans l'arbre ci-dessous, rien n'empêche l'élément `guerrier` de posséder deux enfants `pv`. Ce qui visiblement dans ce contexte, n'aurais aucun sens.



DTD permet donc de décrire un ensemble de règles associer à la structure d'un document XML. On dit que DTD décrit la ***grammaire*** du document.

Chacune de ces règles va décrire le contenu autorisé d'un élément ou l'ensemble des attributs existant pour un élément.

Ces règles sont *généralement* décrites dans un fichier que l'on va associer au document XML pour lequel on souhaite appliquer ces règles.

On utilise alors un programme externe de vérification auquel on fournit le document XML *ainsi que le document DTD* contenant les règles et qui indiquera si oui ou non les règles sont respectées dans le document XML.

Le programme que nous utiliserons est `xmllint`. C'est un programme console très simple d'utilisation.

Liaison d'un DTD à un document XML

Un DTD peut être associé de 3 façons à un document XML :

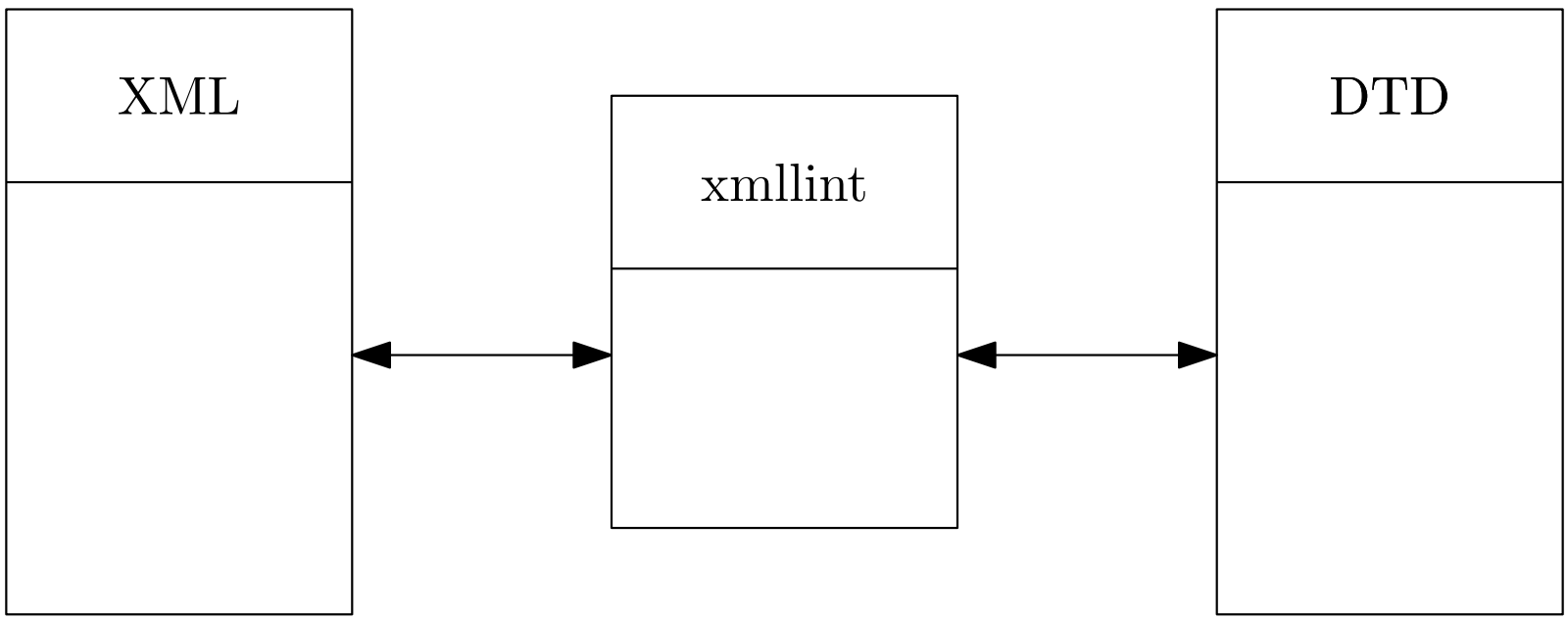
- ***DTD interne*** : toutes les règles sont dans le fichier XML.
- ***DTD externe*** : toutes les règles sont écrire dans un fichier spécifique.
- ***DTD mixte*** : certaines règles sont décrites dans un fichier spécifique et certaines règles sont dans un fichier XML.

Liaison d'un DTD à un document XML

Un DTD peut être associé de 3 façons à un document XML :

- ***DTD interne*** : toutes les règles sont dans le fichier XML.
- ***DTD externe*** : toutes les règles sont écrire dans un fichier spécifique.
- ***DTD mixte*** : certaines règles sont décrites dans un fichier spécifique et certaines règles sont dans un fichier XML.

On utilisera toujours un **DTD externe**.



Déclaration d'un DTD externe

La déclaration d'un DTD externe se fait dans le code source du fichier XML que l'on souhaite associer. On place cette déclaration ***après*** le prologue XML.

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE bonjour SYSTEM "bonjour.dtd">
4
5  <bonjour>Hello World!</bonjour>
```

Cette ligne peut-être lue comme suit : On déclare un ensemble de règles pour la balise `bonjour` donc la description se trouve dans le fichier `bonjour.dtd`. Le chemin du fichier référencé est relatif.

On peut également utiliser une URL pour indiquer le document à utiliser :

```
<!DOCTYPE bonjour SYSTEM "https://www.urldtd.com/bonjour.dtd">
```

Validation d'un fichier XML

À partir du moment où un DTD est associé au document à valider, on peut le valider à l'aide :

- d'un logiciel spécialisé dans le traitement de document XML comme XMLSpy, Eclipse, ...
- d'un programme en utilisant une bibliothèque de fonctions de traitement d'XML, comme par exemple lxml en Python.

Cette validation permet alors de savoir si le document XML respecte les règles indiquées par le DTD, et si ça n'est pas le cas explicite les éléments du fichier XML qui ne sont pas en accord avec les règles.

Structure d'un DTD

Un fichier contient :

- des déclarations d'éléments,
- des déclarations d'attributs,
- des déclarations d'entités,
- éventuellement des commentaires.

Les commentaires ont la même syntaxe que dans un document XML :

```
<!-- Ceci est un commentaire.  -->
```

Chacune de ces déclarations transporte les règles associées. Par exemple, on pourra spécifier que la balise `racine` possède exactement une seule balise enfant dont le nom est `branche`. Chaque type de déclaration possède un jeu de règles qui lui est propre.

Déclaration d'élément

La syntaxe pour la déclaration simple d'un élément est la suivante :

```
<!ELEMENT nom modèle>
```

où

- **ELEMENT** (en majuscule) est un mot clef DTD,
- **nom** doit être un nom valide d'élément (un nom de balise).
- **modèle** est le modèle de contenu de l'élément :
  - ***vide*** l'élément n'a pas de contenu (mais peut avoir des attributs),
  - ***libre*** le contenu de l'élément est un contenu quelconque bien formé,
  - ***données*** l'élément contient du texte,
  - ***éléments*** l'élément est composé d'autres éléments (ses enfants),
  - ***mixte*** l'élément contient un mélange de texte et de sous-éléments.

Déclaration d'élément

La syntaxe pour la déclaration simple d'un élément est la suivante :

```
<!ELEMENT nom modèle>
```

où

- **ELEMENT** (en majuscule) est un mot clef DTD,
- **nom** doit être un nom valide d'élément (un nom de balise).
- **modèle** est le modèle de contenu de l'élément :
  - ***vide*** l'élément n'a pas de contenu (mais peut avoir des attributs),
  - ***libre*** le contenu de l'élément est un contenu quelconque bien formé,
  - ***données*** l'élément contient du texte,
  - ***éléments*** l'élément est composé d'autres éléments (ses enfants),
  - ***mixte*** l'élément contient un mélange de texte et de sous-éléments.

```
<!ELEMENT nom EMPTY>
```

Déclaration d'élément

La syntaxe pour la déclaration simple d'un élément est la suivante :

```
<!ELEMENT nom modèle>
```

où

- **ELEMENT** (en majuscule) est un mot clef DTD,
- **nom** doit être un nom valide d'élément (un nom de balise).
- **modèle** est le modèle de contenu de l'élément :
  - ***vide*** l'élément n'a pas de contenu (mais peut avoir des attributs),
  - ***libre*** le contenu de l'élément est un contenu quelconque bien formé,
  - ***données*** l'élément contient du texte,
  - ***éléments*** l'élément est composé d'autres éléments (ses enfants),
  - ***mixte*** l'élément contient un mélange de texte et de sous-éléments.

```
<!ELEMENT nom ANY>
```

Déclaration d'élément

La syntaxe pour la déclaration simple d'un élément est la suivante :

```
<!ELEMENT nom modèle>
```

où

- **ELEMENT** (en majuscule) est un mot clef DTD,
- **nom** doit être un nom valide d'élément (un nom de balise).
- **modèle** est le modèle de contenu de l'élément :
  - ***vide*** l'élément n'a pas de contenu (mais peut avoir des attributs),
  - ***libre*** le contenu de l'élément est un contenu quelconque bien formé,
  - ***données*** l'élément contient du texte,
  - ***éléments*** l'élément est composé d'autres éléments (ses enfants),
  - ***mixte*** l'élément contient un mélange de texte et de sous-éléments.

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

Déclaration d'élément

La syntaxe pour la déclaration simple d'un élément est la suivante :

```
<!ELEMENT nom modèle>
```

où

- **ELEMENT** (en majuscule) est un mot clef DTD,
- **nom** doit être un nom valide d'élément (un nom de balise).
- **modèle** est le modèle de contenu de l'élément :
  - ***vide*** l'élément n'a pas de contenu (mais peut avoir des attributs),
  - ***libre*** le contenu de l'élément est un contenu quelconque bien formé,
  - ***données*** l'élément contient du texte,
  - ***éléments*** l'élément est composé d'autres éléments (ses enfants),
  - ***mixte*** l'élément contient un mélange de texte et de sous-éléments.

```
<!ELEMENT nom (enfant1, enfant2)>
```

Déclaration d'élément

La syntaxe pour la déclaration simple d'un élément est la suivante :

```
<!ELEMENT nom modèle>
```

où

- **ELEMENT** (en majuscule) est un mot clef DTD,
- **nom** doit être un nom valide d'élément (un nom de balise).
- **modèle** est le modèle de contenu de l'élément :
  - ***vide*** l'élément n'a pas de contenu (mais peut avoir des attributs),
  - ***libre*** le contenu de l'élément est un contenu quelconque bien formé,
  - ***données*** l'élément contient du texte,
  - ***éléments*** l'élément est composé d'autres éléments (ses enfants),
  - ***mixte*** l'élément contient un mélange de texte et de sous-éléments.

```
<!ELEMENT nom (#PCDATA, enfant1, enfant2)>
```

Modèle de contenu d'élément

On définit le contenu à l'aide d'une ***expression régulière*** de sous-éléments :

Modèle de contenu d'élément

On définit le contenu à l'aide d'une ***expression régulière*** de sous-éléments :

- **séquence**

  ```
  <!ELEMENT chapitre (titre, intro, section)>
  ```

  Cela signifie que la balise `chapitre` possède exactement trois balises enfants `titre, intro, section`. ***L'ordre est important!***

Modèle de contenu d'élément

On définit le contenu à l'aide d'une ***expression régulière*** de sous-éléments :

- **séquence**

  ```
  <!ELEMENT chapitre (titre, intro, section)>
  ```

  Cela signifie que la balise `chapitre` possède exactement trois balises enfants `titre`, `intro`, `section`. ***L'ordre est important!***

- **choix**

  ```
  <!ELEMENT chapitre (titre, intro, (section | sections))>
  ```

  Cela signifie que la balise `chapitre` possède exactement trois balises enfants `titre`, `intro` et `section` ***ou*** `sections`.

Modèle de contenu d'élément

On définit le contenu à l'aide d'une ***expression régulière*** de sous-éléments :

- **séquence**

```
<!ELEMENT chapitre (titre, intro, section)>
```

  Cela signifie que la balise `chapitre` possède exactement trois balises enfants `titre`, `intro`, `section`. ***L'ordre est important!***

- **choix**

```
<!ELEMENT chapitre (titre, intro, (section | sections))>
```

  Cela signifie que la balise `chapitre` possède exactement trois balises enfants `titre`, `intro` et `section` ***ou*** `sections`.

- **indicateurs d'occurence**

```
<!ELEMENT chapitre (titre*, intro?, section+)>
```

  - Le symbole * indique un nombre indéterminé d'éléments (entre 0 et n).
  - Le symbole ? indique que l'élément est optionnel (entre 0 et 1).
  - Le symbole + indique la présence d'au moins un élément mais pas de borne supérieure (entre 1 et n).

Modèle de contenu d'élément

On définit le contenu à l'aide d'une ***expression régulière*** de sous-éléments :

- **séquence**

  ```
  <!ELEMENT chapitre (titre, intro, section)>
  ```

  Cela signifie que la balise `chapitre` possède exactement trois balises enfants `titre`, `intro`, `section`. ***L'ordre est important!***
- **choix**

  ```
  <!ELEMENT chapitre (titre, intro, (section | sections))>
  ```

  Cela signifie que la balise `chapitre` possède exactement trois balises enfants `titre`, `intro` et `section` ***ou*** `sections`.
- **indicateurs d'occurence**

  ```
  <!ELEMENT chapitre (titre*, intro?, section+)>
  ```

  - Le symbole * indique un nombre indéterminé d'éléments (entre 0 et n).
  - Le symbole ? indique que l'élément est optionnel (entre 0 et 1).
  - Le symbole + indique la présence d'au moins un élément mais pas de borne supérieure (entre 1 et n).

Remarque : on peut combiner les opérateurs, par exemple :

```
<!ELEMENT texte-section (p|f)*>
```

indique que la balise `texte-section` possède un nombre indéterminé de balises enfants qui peut être soit `p`, soit `f`. L'expression regulière (p|f)* génère un mot (potentiellement vide) composé des lettres ***p*** et ***f***.

Exemple : `pppppffpfpf`, `ffffffff`, `fpfpfppppfffppfp`.

Contenu mixte

Une seule façon de mélanger texte `#PCDATA` et des sous-éléments est acceptée. Il faut placer `#PCDATA` en premier membre d'un choix placé sous une étoile.

```
<!ELEMENT racine (#PCDATA | em | exposant | indice | renvoi)*>
```

La déclaration de la règle ci-dessus autorise un élément racine a avoir entre 0 et *n* enfants qui doivent faire partie la liste `em, exposant, indice, renvoi` avec la particularité que les données brutes (du texte) sont autorisées.

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <racine>
6      <exposant>100</exposant>
7      Texte1
8      <em>balise em</em>
9      <indice>2</indice>
10     Texte2
11     Texte3
12 </racine>
```

Exemple

```
<!ELEMENT catalogue (stage)*>
<!ELEMENT stage (intitule, prerequis?)>
<!ELEMENT intitule (#PCDATA)>
<!ELEMENT prerequis (#PCDATA | xref)*>
<!ELEMENT xref EMPTY>
```

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

<!DOCTYPE document SYSTEM "document.dtd">

<catalogue>
   <stage test="att1">
      <intitule>XML est les bases de données</intitule>
      <prerequis>
         connaitre les langages SQL et HTML
      </prerequis>
   </stage>
   <stage>
      <intitule>XML programmation</intitule>
      <prerequis>
         avoir suivi le cours de Bases de données et Web
      </prerequis>
   </stage>
</catalogue>
```

Déclaration d'attributs

```
<!ATTLIST element nom-attribut1 type1 defaut1 nom-attribut2 type2 defaut2 ...>
```

Le type d'un attribut définit les valeurs qu'il peut prendre :

- **CDATA** : valeur chaîne de caractères,
- **ID**, **IDREF**, **IDREFS** permettent de définir des références à l'intérieur d'un document,

Remarque : il n'y a pas de # devant le CDATA, et c'est bien le mot-clef **CDATA** et non pas PCDATA.

Déclaration d'attributs

La déclaration par défaut peut prendre quatre formes :

- la valeur par défaut de l'attribut (choisie par la personne qui écrit le document DTD),
- **#REQUIRED** indique que l'attribut est obligatoire,
- **#IMPLIED** indique que l'attribut est optionnel,
- **#FIXED valeur** (où **valeur** est une valeur choisie par la personne qui écrit le document DTD) indique que l'attribut prend toujours la même valeur, dans toute instance de l'élément ***si l'attribut y apparaît***.

Il faudra toujours penser à spécifier cet option pour chacun des attributs déclarés. Donner le nom de l'attribut et son type ne suffit pas, il faut également indiquer le défaut.

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document version="1.0">
2     ...
3  </document>
```

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document version="1.0">
2     ...
3  </document>
```

VALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document version="1.0">
2     ...
3  </document>
```

VALIDE

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document version="2.0">
2     ...
3  </document>
```

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document version="1.0">
2     ...
3  </document>
```

VALIDE

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document version="2.0">
2     ...
3  </document>
```

VALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document>
2     ...
3  </document>
```

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document>
2     ...
3  </document>
```

VALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document>
2     ...
3  </document>
```

VALIDE

```
<!ATTLIST document version CDATA #FIXED "1.0">
```

```
1  <document version="1.0">
2     ...
3  </document>
```

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA "1.0">
```

```
1  <document>
2     ...
3  </document>
```

VALIDE

```
<!ATTLIST document version CDATA #FIXED "1.0">
```

```
1  <document version="1.0">
2     ...
3  </document>
```

VALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA #FIXED "1.0">
```

```
1  <document version="2.0">
2     ...
3  </document>
```

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA #FIXED "1.0">
```

```
1  <document version="2.0">
2     ...
3  </document>
```

INVALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA #FIXED "1.0">
```

```
1  <document version="2.0">
2     ...
3  </document>
```

INVALIDE

```
<!ATTLIST document version CDATA #FIXED "1.0">
```

```
1  <document>
2     ...
3  </document>
```

Exemples de déclaration d'attributs

```
<!ATTLIST document version CDATA #FIXED "1.0">
```

```
1 <document version="2.0">
2    ...
3 </document>
```

INVALIDE

```
<!ATTLIST document version CDATA #FIXED "1.0">
```

```
1 <document>
2    ...
3 </document>
```

VALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="Mme" nom-epouse="Lenoir">
2     Martin
3  </nom>
```

Exemples de déclaration d'attributs

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="Mme" nom-epouse="Lenoir">
2     Martin
3  </nom>
```

VALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="Mme" nom-epouse="Lenoir">
2     Martin
3  </nom>
```

VALIDE

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="M." nom-epouse="Lenoir">
2     Martin
3  </nom>
```

Exemples de déclaration d'attributs

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="Mme" nom-epouse="Lenoir">
2     Martin
3  </nom>
```

VALIDE

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="M." nom-epouse="Lenoir">
2     Martin
3  </nom>
```

VALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="M.">
2     Martin
3  </nom>
```

Exemples de déclaration d'attributs

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="M.">
2     Martin
3  </nom>
```

VALIDE

Exemples de déclaration d'attributs

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="M.">
2     Martin
3  </nom>
```

VALIDE

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="Madame" nom-epouse="Lenoir">
2     Martin
3  </nom>
```

Exemples de déclaration d'attributs

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="M.">
2     Martin
3  </nom>
```

VALIDE

```
<!ATTLIST nom titre (Mlle|Mme|M.) #REQUIRED nom-epouse CADATA #IMPLIED>
```

```
1  <nom titre="Madame" nom-epouse="Lenoir">
2     Martin
3  </nom>
```

INVALIDE

Attributs ID, IDREF, IDREFS

Un attribut **ID** sert à référencer un élément, la valeur de cette référence pouvant être rappelée dans des attributs **IDREF** ou **IDREFS**.

Un élément ne peut avoir au plus qu'un attribut **ID** et la valeur associée doit être unique dans le document XML. Cette valeur doit être un ***nom*** XML (donc pas un nombre, par exemple).

La valeur de défaut pour un attribut **ID** est obligatoirement **#REQUIRED** ou **#IMPLIED**.

Une valeur utilisée dans un attribut **IDREF** ou **IDREFS** doit obligatoirement correspondre à celle d'un attribut **ID**.

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREF #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6      <personne id="id-1">
7          <nom>Dupond</nom>
8          <prenom>Martin</prenom>
9      </personne>
10
11     <personne id="id-2">
12         <nom>Durand</nom>
13         <prenom>Helmut</prenom>
14     </personne>
15
16     <livre auteur="id-1">Ma vie, mon oeuvre</livre>
17 </document>
```

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREF #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6      <personne id="id-1">
7          <nom>Dupond</nom>
8          <prenom>Martin</prenom>
9      </personne>
10
11     <personne id="id-2">
12         <nom>Durand</nom>
13         <prenom>Helmut</prenom>
14     </personne>
15
16     <livre auteur="id-1">Ma vie, mon oeuvre</livre>
17 </document>
```

VALIDE

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREF #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6     <personne id="id-1">
7        <nom>Dupond</nom>
8        <prenom>Martin</prenom>
9     </personne>
10
11    <personne id="id-1">
12       <nom>Durand</nom>
13       <prenom>Helmut</prenom>
14    </personne>
15
16    <livre auteur="id-1">Ma vie, mon oeuvre</livre>
17 </document>
```

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREF #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6     <personne id="id-1">
7        <nom>Dupond</nom>
8        <prenom>Martin</prenom>
9     </personne>
10
11    <personne id="id-1">
12       <nom>Durand</nom>
13       <prenom>Helmut</prenom>
14    </personne>
15
16    <livre auteur="id-1">Ma vie, mon oeuvre</livre>
17 </document>
```

INVALIDE

Exemples ID, IDREF, IDREFS

```
<!ELEMENT document (personne*, livre*)>
<!ELEMENT personne (nom, prenom)>
<!ATTLIST personne id ID #REQUIRED>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT prenom (#PCDATA)>
<!ELEMENT livre (#PCDATA)>
<!ATTLIST livre auteur IDREF #IMPLIED>
```

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

<!DOCTYPE document SYSTEM "document.dtd">

<document>
    <personne id="id-1">
        <nom>Dupond</nom>
        <prenom>Martin</prenom>
    </personne>

    <personne id="id-2">
        <nom>Durand</nom>
        <prenom>Helmut</prenom>
    </personne>

    <livre auteur="id-3">Ma vie, mon oeuvre</livre>
</document>
```

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREF #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6      <personne id="id-1">
7          <nom>Dupond</nom>
8          <prenom>Martin</prenom>
9      </personne>
10
11     <personne id="id-2">
12         <nom>Durand</nom>
13         <prenom>Helmut</prenom>
14     </personne>
15
16     <livre auteur="id-3">Ma vie, mon oeuvre</livre>
17 </document>
```

INVALIDE

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREF #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6      <personne id="id-1">
7          <nom>Dupond</nom>
8          <prenom>Martin</prenom>
9      </personne>
10
11     <personne id="id-2">
12         <nom>Durand</nom>
13         <prenom>Helmut</prenom>
14     </personne>
15
16     <livre auteur="id-1 id-2">Ma vie, mon oeuvre</livre>
17 </document>
```

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREF #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6     <personne id="id-1">
7        <nom>Dupond</nom>
8        <prenom>Martin</prenom>
9     </personne>
10
11    <personne id="id-2">
12       <nom>Durand</nom>
13       <prenom>Helmut</prenom>
14    </personne>
15
16    <livre auteur="id-1 id-2">Ma vie, mon oeuvre</livre>
17 </document>
```

INVALIDE

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREFS #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6      <personne id="id-1">
7          <nom>Dupond</nom>
8          <prenom>Martin</prenom>
9      </personne>
10
11     <personne id="id-2">
12         <nom>Durand</nom>
13         <prenom>Helmut</prenom>
14     </personne>
15
16     <livre auteur="id-1 id-2">Ma vie, mon oeuvre</livre>
17 </document>
```

Exemples ID, IDREF, IDREFS

```
1  <!ELEMENT document (personne*, livre*)>
2  <!ELEMENT personne (nom, prenom)>
3  <!ATTLIST personne id ID #REQUIRED>
4  <!ELEMENT nom (#PCDATA)>
5  <!ELEMENT prenom (#PCDATA)>
6  <!ELEMENT livre (#PCDATA)>
7  <!ATTLIST livre auteur IDREFS #IMPLIED>
```

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE document SYSTEM "document.dtd">
4
5  <document>
6      <personne id="id-1">
7          <nom>Dupond</nom>
8          <prenom>Martin</prenom>
9      </personne>
10
11     <personne id="id-2">
12         <nom>Durand</nom>
13         <prenom>Helmut</prenom>
14     </personne>
15
16     <livre auteur="id-1 id-2">Ma vie, mon oeuvre</livre>
17 </document>
```

VALIDE

Déclaration des entités

Il existe trois types d'**entités** :

- ***Les entités internes*** : ce sont des *macros exportées* qui sont utilisées dans le document XML validé par le DTD.
- ***Les entités paramétriques*** : ce sont des *macros non exportées* qui sont utilisées ailleurs ***dans*** le DTD.
- ***Les entités externes*** : ce sont des *macros importées*, définies dans un autre document, utilisable dans le DTD lui-même ou dans tout document XML valide pour le DTD.

Les entités permettent de simplifier l'écriture du DTD et du XML en utilisant des macros, par exemple pour des symboles spéciaux, des longs textes, etc ...

Exemples d'entités

- **Entité interne** : `<!ENTITY euro "&#8364;">` permet de créer une macro pour le symbole euro. Dans le document XML on utilisera la macro `&euro;`.
- **Entité paramétrique** : `<!ENTITY % editeur "O'Reilly">` permet de créer une macro pour le texte *"O'Reilly"*. Dans le document DTD on utilisera la macro `%editeur;`.
- **Entité externe** : `<!ENTITY % euro SYSTEM "fichier-entite.dtd">` permet d'utiliser la macro `euro` référencée dans le fichier `fichier-entite.dtd`.

Exemples d'entités

```
1  <!--
2      Entité externe pour importer les entités
3      représentant les caractères accentués
4   -->
5  <!ENTITY % HTMLlat1 PUBLIC
6  "-//W3C//ENTITIES Latin 1 for XHTML//EN"
7  "http://www.w3.org/TR/xhtml1/DTD/xhtml-lat1.ent">
8
9  <!-- Ceci reviens à faire un import. -->
10 %HTMLlat1;
11
12 <!-- Entité paramétrique -->
13 <!ENTITY % elt "(#PCDATA|elt1)*" >
14 <!ELEMENT racine %elt;>
15 <!ELEMENT elt1 (#PCDATA)>
16
17 <!--Entités interne -->
18 <!ENTITY euro "&#8364;">
19 <!ENTITY LILLE1 "Universit&eacute; Lille 1">
20
21 <!--
22     L'utilisation du &eacute; est possible car
23     c'est une entité externe importée à l'aide de %HTMLlat1
24 -->
```

Exemples d'entités

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE racine SYSTEM "./entites.dtd">
4
5  <racine>
6     blabla
7     <elt1>
8        Universit&eacute; : &LILLE1;
9     </elt1>
10    <elt1>
11       10000 &euro;
12    </elt1>
13 </racine>
```

1. On utilise le caractère spécial ***accent aigu*** grâce à la macro `&eacute;` que l'on a importé via `%HTMLlat1`. On utilise également la macro `&LILLE1;` que l'on a créé dans le DTD.
2. On utilise la macro `&euro;` que l'on a créé dans le DTD. Cette macro sera interprété comme le symbole euro à la lecture du XML par un programme externe.

Exemples d'entités

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE racine SYSTEM "./entites.dtd">
4
5  <racine>
6     blabla
7     <elt1>
8        Universit&eacute; : &LILLE1;
9     </elt1>
10    <elt1>
11       10000 &euro;
12    </elt1>
13 </racine>
```

1. On utilise le caractère spécial ***accent aigu*** grâce à la macro `&eacute;` que l'on a importé via `%HTMLlat1`. On utilise également la macro `&LILLE1;` que l'on a créé dans le DTD.
2. On utilise la macro `&euro;` que l'on a créé dans le DTD. Cette macro sera interprété comme le symbole euro à la lecture du XML par un programme externe.

Exemples d'entités

```
1  <?xml version="1.0" encoding="utf-8"?>
2
3  <!DOCTYPE racine SYSTEM "./entites.dtd">
4
5  <racine>
6     blabla
7     <elt1>
8        Universit&eacute; : &LILLE1;
9     </elt1>
10    <elt1>
11       10000 &euro;
12    </elt1>
13 </racine>
```

1. On utilise le caractère spécial ***accent aigu*** grâce à la macro `&eacute;` que l'on a importé via `%HTMLlat1`. On utilise également la macro `&LILLE1;` que l'on a créé dans le DTD.
2. On utilise la macro `&euro;` que l'on a créé dans le DTD. Cette macro sera interprété comme le symbole euro à la lecture du XML par un programme externe.

Quizz

Rédiger une DTD pour une bibliographie. Cette bibliographie :

- Contient des livres et des articles.
- Les informations nécessaires pour un livre sont :
  - Son titre général.
  - Les noms des auteurs.
  - Ses tomes et pour chaque tome, leur nombre de pages.
  - Des informations générales sur son édition comme par exemple le nom de l'éditeur, le lieu d'édition, le lieu d'impression, son numéro ISBN.
- Les informations nécessaires pour un article sont :
  - Son titre.
  - Les noms des auteurs.
  - Ses références de publication : nom du journal, numéro des pages, année de publication et numéro du journal.
- On réservera aussi un champ optionnel pour un avis personnel.

Quizz

```
1  <!ELEMENT biblio (livre|article)*>
2  <!ELEMENT livre (titre, auteur+, tome*, edition, avis?)>
3  <!ELEMENT titre (#PCDATA)>
4  <!ELEMENT auteur (#PCDATA)>
5  <!ELEMENT tome (nb_pages)>
6  <!ELEMENT nb_pages (#PCDATA)>
7  <!ELEMENT edition (editeur, lieu_edition, lieu_impression, isbn)>
8  <!ELEMENT editeur (#PCDATA)>
9  <!ELEMENT lieu_edition (#PCDATA)>
10 <!ELEMENT lieu_impression (#PCDATA)>
11 <!ELEMENT isbn (#PCDATA)>
12 <!ELEMENT avis (#PCDATA)>
13 <!ELEMENT article (titre, auteur+, journal)>
14 <!ELEMENT journal (nom_journal, page, num_journal, annee)>
15 <!ELEMENT nom_journal (#PCDATA)>
16 <!ELEMENT page (#PCDATA)>
17 <!ELEMENT num_journal (#PCDATA)>
18 <!ELEMENT annee (#PCDATA)>
```

Quizz

Modifier la DTD précédente :

- En ajoutant un attribut optionnel soustitre à l'élément titre.
- En faisant de l'élément tome un élément vide et en lui ajoutant un attribut requis nb_pages et un attribut optionnel soustitre.
- En faisant de l'élément nom_journal un attribut de l'élément journal et en lui donnant comme valeur par défaut Feuille de Chou.
- En faisant de l'élément annee un attribut de type énuméré, prenant comme valeurs possibles 2000, 2001, 2002, "avant_2000" et "inconnue" et proposant comme valeur par défaut inconnue.

Quizz

```
1  <!ELEMENT biblio (livre|article)*>
2  <!ELEMENT livre (titre, auteur+, tome*, edition, avis?)>
3  <!ATTLIST titre soustitre CDATA #IMPLIED>
4  <!ELEMENT titre (#PCDATA)>
5  <!ELEMENT auteur (#PCDATA)>
6  <!ELEMENT tome EMPTY>
7  <!ATTLIST tome nb_pages CDATA #REQUIRED soustitre CDATA #IMPLIED>
8  <!ELEMENT edition (editeur, lieu_edition, lieu_impression, isbn)>
9  <!ELEMENT editeur (#PCDATA)>
10 <!ELEMENT lieu_edition (#PCDATA)>
11 <!ELEMENT lieu_impression (#PCDATA)>
12 <!ELEMENT isbn (#PCDATA)>
13 <!ELEMENT avis (#PCDATA)>
14 <!ELEMENT article (titre, auteur+, journal)>
15 <!ELEMENT journal (page, num_journal)>
16 <!ATTLIST journal nom_journal CDATA "Feuille de Chou" annee
17         (2000 | 2001 | 2002 | avant_2000 | inconnue) "inconnue">
18 <!ELEMENT page (#PCDATA)>
19 <!ELEMENT num_journal (#PCDATA)>
20 <!ELEMENT annee (#PCDATA)>
```