

# **Définition de Type de Document**

**Ramzi MAHMOUDI**

**Cours adapté du travail de : Pierre Alain Renyier, Gilles Chagnon , Rémi Eyraud et Silvano Dal Zilio**

## Plan du cours

- DTD : Définition, Utilisation
- DTD : Syntaxe des éléments
- DTD : Syntaxe des attributs
- DTD : Notions d'entités
- DTD : Limitations

# DTD – Définition, Utilisation

## Définition

Un document XML peut être associé à :

- une DTD ou un schéma XML pour décrire les balises autorisées (et leur structure)
- une feuille de style pour présenter

les données DTD et schéma XML permettent de définir son propre langage basé sur XML :

- Vocabulaire (balises)
- Grammaire (règles d'imbrication)

-> Dialecte ou Jargon

# DTD – Définition, Utilisation

## Définition

- Les DTD sont des fichiers spécifiant le **vocabulaire** et la **structure** d'un type de documents XML.
- But : rajouter des contraintes à la forme des éléments et attributs (nom, ordre, contenu).
- Un document XML *bien formé suivant une DTD* donnée est appelé **document valable**.
- Supporté par la plupart des logiciels (java, firefox, ...)
- Ce n'est pas du XML.
- On définit rarement une DTD à partir de rien.

# DTD – Définition, Utilisation

## Utilisation d'une DTD Externe

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>  
<!DOCTYPE baliseracine SYSTEM "http://www.isim.com/test.dtd">
```

- standalone="no" : pour dire au logiciel d'aller chercher une DTD externe.
- baliseracine : nom de la balise racine définie dans la DTD.
- http://etc. : emplacement (global ou local) où trouver la DTD.

# DTD – Définition, Utilisation

## Utilisation d'une DTD Interne

```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE baliseracine [
<!ELEMENT baliseracine (PERSONNE*)>
<!ELEMENT PERSONNE (#PCDATA)>
<!ATTLIST PERSONNE PNUM ID #REQUIRED>
]>
<baliseracine>
</baliseracine>
```

- standalone="yes" : pour dire au logiciel que la DTD est définie dans le document
- DOCTYPE : mot-clé correspondant au début de la définition de la DTD.

# DTD – Définition, Utilisation

## DTD externe, interne

- **Externe** : en spécifiant l'adresse de la DTD.
- **Interne** : directement en entête du fichier XML.
- **Combinaison des deux** :

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>  
<!DOCTYPE problemset SYSTEM "../dtd/extdtd.dtd"[  
<!ELEMENT question #PCDATA>  
...  
>
```

La DTD interne est lue avant la DTD externe, il ne doit pas y avoir de conflit.

# DTD : Syntaxe des éléments

- **<!ELEMENT balise (définition)>** permet de définir ce qui peut être contenu dans un élément de nom “balise”.
- Le paramètre *définition* représente soit un type de donnée prédéfini, soit un élément de données composé, constitué lui-même d'éléments
- Types prédéfinis :
  - **ANY** : L'élément peut contenir tout type de donnée
  - **EMPTY** : L'élément ne contient pas de données spécifiques
  - **#PCDATA** : L'élément doit contenir une chaîne de caractère
- Utilisation :
  - ANY et EMPTY : sans parenthèses
  - (#PCDATA) : avec parenthèses

# DTD : Syntaxe des éléments

Déclaration d'éléments composés :

Exemple : **<!ELEMENT montant (devise?,valeur) >**

| Opérateur | Signification   | Exemple |
|-----------|---|---------|
| +         | L'élément doit être présent au minimum une fois   | A+      |
| *         | L'élément peut être présent plusieurs fois (ou aucune)  | A*      |
| ?         | L'élément peut être optionnellement présent   | A?      |
|           | L'élément A <b>ou</b> B peuvent être présents (pas les deux)                                      | A B     |
| ,         | L'élément A doit être présent et suivi de l'élément B   | A,B     |
| ()        | Les parenthèses permettent de regrouper des éléments afin de leur appliquer les autres opérateurs | (A,B)+  |

Attention à l'ordre !

**<!ELEMENT mont (devise, somme) > ≠ <!ELEMENT mont (somme, devise) >**

# DTD : Syntaxe des éléments

## Exemple de définition d'éléments

- Exemple de DTD :
  - <!ELEMENT personne (nom, prenom+, tel?, adresse) >
  - <!ELEMENT nom (#PCDATA) >
  - <!ELEMENT prenom (#PCDATA) >
  - <!ELEMENT tel (#PCDATA) >
  - <!ELEMENT email (#PCDATA) >
  - <!ELEMENT adresse ANY >

# DTD : Syntaxe des éléments

## Exemple de définition d'éléments

- Exemple de document :

```
< personne >
  < nom >Hugo< /nom >
  < prenom >Victor< /prenom >
  < prenom >Charles< /prenom >
  < tel >01120243< /tel >
  < adresse >< rue >< /rue >< ville >Paris< /ville >< /adresse >
< / personne >
```

# DTD : Syntaxe des attributs

- Par défaut, aucun attribut aux éléments.
- **<!ATTLIST nom\_element nom\_attribut type mode>**
- Le mode précise si l'attribut est obligatoire (#REQUIRED) ou optionnel (#IMPLIED), ou fixé (#FIXED).

<!ATTLIST homme age CDATA #IMPLIED> signifie qu'un élément "homme" peut avoir un attribut "age" textuel.

<!ATTLIST homme age CDATA #FIXED "40 ans">

<!ATTLIST piece position (**face | pile**) #REQUIRED>

- Pour associer plusieurs attributs à un élément :
  - spécifier plusieurs lignes "<!ATTLIST ...>"
  - ou en mettre plusieurs à la fois (ordre sans importance).

# DTD : Syntaxe des attributs

## Types des Attributs

Le type de valeur peut être :

- **CDATA** : texte,
- **ID** : nom XML (et non pas token de nom), identifiant l'élément dans le document. La valeur d'un attribut de type ID ne peut pas être utilisée deux fois dans le même document et il y a au plus un attribut de type ID par élément
- **IDREF** ou **IDREFS** : identificateur ou suite d'identificateurs (séparés par une suite de séparateurs : espace, CR, LF, tabulation),
- **NMTOKEN** ou **NMTOKENS** : token de nom ou suite de tokens de nom, (*nom<sub>1</sub> ... nom<sub>n</sub>*)
- **Enumeration**: (*nom<sub>1</sub> | ... | nom<sub>n</sub>*)

# DTD : Syntaxe des attributs

## Exemples d'attributs

```
<! ATTLIST personne
    num ID                #REQUIRED
    age CDATA             #IMPLIED
    genre (Masculin | Feminin ) #REQUIRED
>

<!ELEMENT auteur (#PCDATA) >
<!ELEMENT editeur (#PCDATA) >

<!ATTLIST auteur
    genre (Masculin | Feminin ) #REQUIRED
    ville CDATA                 #IMPLIED
>

<!ATTLIST editeur
    ville CDATA #FIXED "Paris">
```

# DTD : Syntaxe des attributs

## Exemple de ID et IDREF

```
<?xml version="1.0" standalone="yes"?>  
<!DOCTYPE DOCUMENT [  
<!ELEMENT DOCUMENT(PERSONNE*)>  
<!ELEMENT PERSONNE (#PCDATA)>  
<!ATTLIST PERSONNE PNUM ID #REQUIRED>  
<!ATTLIST PERSONNE MERE IDREF #IMPLIED>  
<!ATTLIST PERSONNE PERE IDREF #IMPLIED>  
>]
```

```
<DOCUMENT>  
<PERSONNE PNUM = "P1">Marie</PERSONNE>  
<PERSONNE PNUM = "P2">Jean</PERSONNE>  
<PERSONNE PNUM = "P3" MERE="P1" PERE="P2">Pierre</PERSONNE>  
<PERSONNE PNUM = "P4" MERE="P1" PERE="P2">Julie</PERSONNE>  
</DOCUMENT>
```

## DTD - Notion d'entités

```
<!ENTITY % mon_entite "#PCDATA|age*">
```

=> Cela signifie que partout “%mon\_entite” sera remplacé par “#PCDATA|age\*”

- Les **entités paramètres** doivent être définies dans une DTD externe (elles ne sont pas prises en considération si elles sont définies en interne).

# DTD - Notion d'entités

## Quelques règles d'écriture

- **Modularité**
  - définir dans des entités séparées les parties réutilisables
- **Précédence**
  - Regrouper les déclarations d'entités en tête
- **Abstraction**
  - Utiliser des entités pour les modèles de contenus
- **Spécificité**
  - Éviter les DTD trop générales
- **Simplicité**
  - Découper les DTD trop complexes

# DTD - Limitations

- Pas de types de données
  - difficile à interpréter par le récepteur
  - difficile à traduire en schéma objets
- Pas en XML
  - langage spécifique
- Propositions de compléments
  - XML-data de Microsoft (BizTalk)
  - **XML-schema du W3C**