

# DTD

## Définition de Type de documents

1. Document valide par rapport à une DTD
2. Structure d'une DTD
3. Déclaration d'un élément
4. Déclaration d'un attribut
5. Déclaration d'entités
6. DTD conditionnelles
7. Déclaration d'entités non interprétables

### Références:

XML La synthèse, Intégrer XML dans vos architectures, A.Boukhors and al. Eds Dunod

Xml in a nutshell, Manuel de référence, eds O'Reilly

<http://www.w3schools.com/>

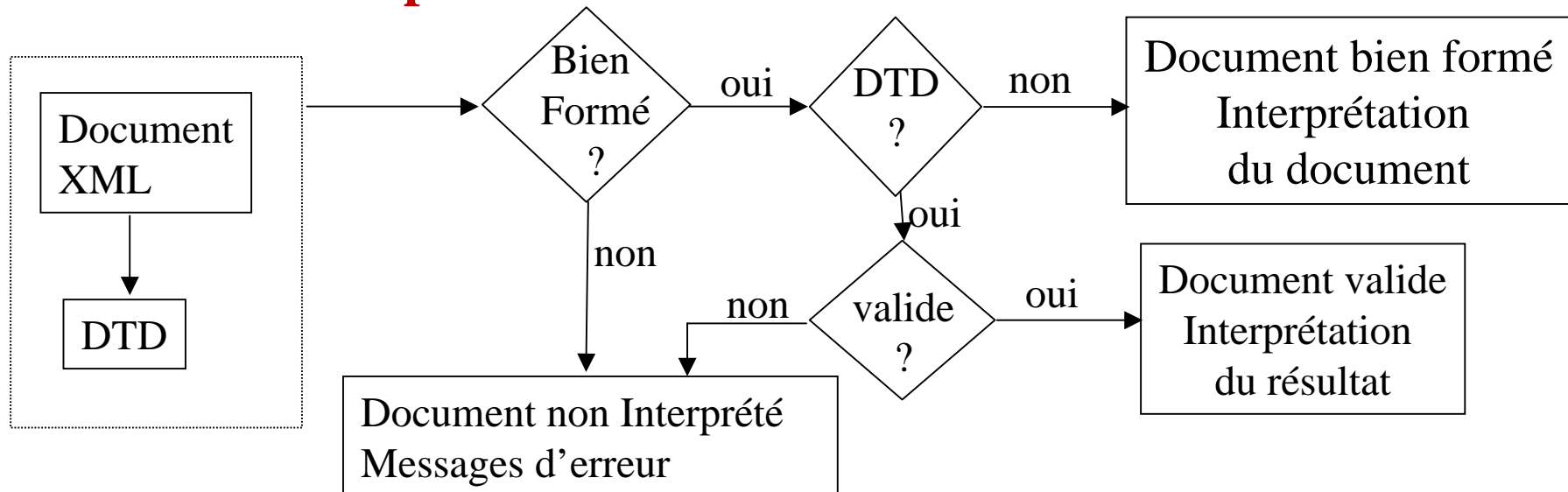
## Document bien formé - Document valide

### Document bien formé

**Un document est syntaxiquement correct et peut être traduit sans erreur par tout navigateur ou interpréteur XML.**

### Document valide

**Un document est valide par rapport à une DTD, c'est-à-dire que son contenu respecte les définitions contenues dans la DTD.**



## Déclaration de type de document

### Introduit la définition de type de documents

- **Propriété**

Elément du prologue

- **Notation**

Déclaration de DTD interne → déclaration pour tout document ayant comme élément racine : *nom\_racine*

```
<!DOCTYPE nom_racine [dtd_interne]>  
< nom_racine > contenu </ nom_racine >
```

- **Exemple**

```
<?xml version= "1.1" standalone= "yes" ?>  
<!DOCTYPE test [<!ELEMENT test ANY>]>  
<test> le contenu peut être de n'importe quel format </test>
```

## Déclaration externe

- Déclaration de DTD externe

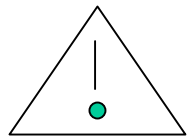
**<!DOCTYPE nom\_racine SYSTEM "racine.dtd " >**

**< nom\_racine > contenu </nom\_racine >**

- Déclaration de DTD externe + déclaration Interne

**<!DOCTYPE nom\_racine SYSTEM "racine.dtd "[dtd\_interne]>**

**< nom\_racine > contenu </nom\_racine >**



Les éléments de la DTD interne sont évalués en premier  
Et reste donc valides sur tout le document.

⇒ Pas de redéfinition d'un élément entre DTD externe  
et interne.

## Exemple de déclaration externe et interne de DTD

```
<?xml version= "1.0" standalone= "no" ?>
<!DOCTYPE personne SYSTEM "nom.dtd"
[<!ELEMENT profession ANY >
<!ELEMENT personne (nom, profession) >
] >
<!--un élément fils nom et un élément fils
    profession-->
<personne>
  <nom>
    <prenom> Gaston </prenom>
    <famille> Lagaffe </famille>
  </nom>
  <profession>archiviste </profession>
</personne>
```

```
<!--DTD pour nom.dtd -->
<!ELEMENT nom (prenom, famille)>
<!ELEMENT prenom ANY >
<!ELEMENT famille ANY >
```

DTD interne déclare  
les éléments *personne*  
et *profession* mais repose  
sur le fichier *nom.dtd* pour  
contenir la déclaration de  
l'élément *nom*

## Contenu d'une DTD

- Déclaration d'éléments
- Déclaration d'attributs
- Déclaration d'entités paramètres
- Déclaration de notation

# Déclaration d'éléments

**Définit un type d'élément en associant un nom de type à un modèle de contenu**

## Propriété

Tout élément ayant le même nom que le type d'élément doit avoir un contenu conforme au modèle défini dans la déclaration

## Notation

`<!ELEMENT nom_type modèle>`

## Règles de syntaxe

- Premier caractère → alphabétique ou un sous-ligné ( \_ )
- Autres caractères → alphanumérique  
+ sous-ligné ( \_ ) + tiré ( - ) + point ( . )
- Respect de la casse → minuscule et majuscule différenciées
- Pas de blanc
- Xml et XML usage normalisé

## Exemple de déclaration d'éléments

<!DOCTYPE test [

<!ELEMENT test (contenu)>

<!ELEMENT contenu ANY>

]>

<test> <contenu>ceci est un document dont le contenu est  
quelconque </contenu> </test>



## Composants d'une déclaration d'éléments

Un modèle associé à une déclaration d'éléments comprend

- Modèle libre
- Modèle spécifiant des données représentées par un flot de caractères
- Séquence d'un ou n éléments fils
- Choix d'éléments fils
- Modèle mixte spécifiant un modèle de Données + des éléments fils
- Modèle vide

Modèle libre  
ANY

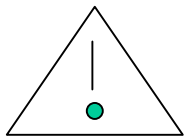
**Spécifie un contenu n'ayant pas de modèle particulier  
Mais qui doit être bien formé**

- **Notation**

<!ELEMENT nom\_type ANY>

- **Propriété**

Utilisé lors de l'élaboration partielle d'une DTD sur les éléments dont le modèle n'est pas encore bien défini.



Si dans un document, l'élément associé au modèle libre admet des éléments fils, ces derniers doivent avoir été déclarés dans la DTD.

## Modèle #PCDATA Parsable Character Data

**Spécifie qu'un élément ne doit contenir que des données textuelles**

- **Notation**

<!ELEMENT nom\_type (#PCDATA)>

- **Propriété**

Ne peut pas contenir de sous élément de quelque type que se soit (images, données non textuelles etc...)

- **Exemple**

```
<!DOCTYPE test [
```

```
<!ELEMENT test (contenu)>
```

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

```
]>
```

```
<test> <contenu>ceci est un document dont le contenu ne peut contenir que des  
données textuelles analysables </contenu> </test>
```

## section littérale

**Une section littérale est définie dans un modèle de type  
#PCDATA**

### Exemple

```
<!DOCTYPE test [  
<!ELEMENT test (contenu)>  
<!ELEMENT contenu (#PCDATA)>  
<test> <contenu> l'élément déclaré  
<![CDATA[<!ELEMENT contenu (#PCDATA)> ]]> peut  
    contenir une section littérale  
</contenu> </test>
```

# Séquence d'éléments fils

**Spécifie suivant un ordre fixe les éléments fils**

- **Notation**

<!ELEMENT nom\_type (nom\_fils\_1, ..., nom\_fils\_n)>

- **Exemple**

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

<!ELEMENT titre (#PCDATA)>

<!ELEMENT intro (#PCDATA)>

<!ELEMENT section ANY>

<chapitre> <titre> Séquence d'éléments fils </titre>

    <intro>

        cette introduction ne peut être placée avant le titre du chapitre ou après l'élément section.

    </intro>

    <section> la section n'a pas encore son modèle de contenu spécifié.

    </section>

</chapitre>

## Séquence: Exemple NON VALIDE

```
<!DOCTYPE nom [ <!ELEMENT nom (prenom, famille)>
<!ELEMENT famille (#PCDATA)>
<!ELEMENT prenom (#PCDATA)>
]>
<nom>
<!--l'élément famille doit être placé après l'élément prenom-->
  <famille> Lagaffe</famille>
  <prenom> Gaston </prenom>
</ nom >
```

## Indicateur du nombre de sous éléments

**spécifie le nombre de fois où un élément fils peut apparaître**

**? → autorise zéro ou un élément**

**\* → autorise 0 ou plusieurs éléments**

**+ → autorise un ou plusieurs éléments**

- **Notation**

**nom\_fils ?**

**nom\_fils \***

**nom\_fils +**

## Indicateurs: exemple valide

```
<?xml version= "1.1" encoding= "ISO-8859-1" standalone= "yes" ?>
<!DOCTYPE chapitre [
  <!ELEMENT chapitre (titre, intro?, section+)>
  <!ELEMENT titre (#PCDATA)>
  <!ELEMENT intro (#PCDATA)>
  <!ELEMENT section ANY>
]>
<chapitre> <titre> chapitre sans intro </titre>
    <section> première section obligatoire.</section>
    <section> deuxième section.</section>
</chapitre>
```



## Indicateurs: exemple non valide

```
<?xml version= "1.1" encoding= "ISO-8859-1" standalone= "yes" ?>
<!DOCTYPE chapitre [
  <!ELEMENT chapitre (titre, sous_titre?, intro, section+)>
  <!ELEMENT sous_titre (#PCDATA)>
  <!ELEMENT titre (#PCDATA)>
  <!ELEMENT intro (#PCDATA)>
  <!ELEMENT section ANY>]>
<chapitre>
  <titre> chapitre avec indicateurs </titre>
    <sous_titre> chapitre avec sous titre </sous_titre>
    <sous_titre> erreur car un seul sous titre est permis </sous_titre>
    <intro> toujours faire attention à l'ordre fixe des balises</intro>
    <section> première section toujours présente. </section>
    <section> deuxième section.</section> </chapitre>
```

## Choix d'éléments fils

**spécifie que le contenu d'un élément peut contenir soit une sorte d'éléments fils ou une autre sorte mais jamais les deux à la fois.**

- **Notation**

`<!ELEMENT nom_type (nom_fils_1| ...| nom_fils_n)>`

- **Exemples**

`<!ELEMENT traitement (param| faute)>`

`<!ELEMENT chiffre (zero| un | deux | trois | quatre | cinq | six | sept | huit |  
neuf )>`

## choix avec indicateurs

```
<!DOCTYPE chapitre [  
  <!ELEMENT chapitre (titre, (sous_titre|intro), section+)>  
  <!ELEMENT section (titre_section, intro_section?, (figure|corps_section)*)>  
  <!ELEMENT sous_titre (#PCDATA)>  
  <!ELEMENT titre (#PCDATA)>  
  <!ELEMENT intro (#PCDATA)>  
  <!ELEMENT titre_section (#PCDATA)>  
  <!ELEMENT intro_section (#PCDATA)>  
  <!ELEMENT corps_section (#PCDATA)>  
  <!ELEMENT figure ANY>]>
```

## Indicateurs sur un modèle de contenu

**Un indicateur peut s'appliquer à une séquence ou un choix d'éléments fils**

```
<!ELEMENT chapitre (titre, intro, section+,exercice*)>  
<!ELEMENT titre (#PCDATA)>  
<!ELEMENT intro (#PCDATA)>  
<!ELEMENT section (titre_section, corps_section)>  
<!ELEMENT exercice ANY>
```

≠

```
<!ELEMENT chapitre (titre, intro, (titre_section, corps_section)+, exercice*)>  
<!ELEMENT titre (#PCDATA)>  
<!ELEMENT intro (#PCDATA)>  
<!ELEMENT exercice ANY>
```

# Parenthèses

**Permet de combiner de manière complexe n'importe quel modèle de contenu**

## Propriétés

1. Un choix ou une séquence peuvent être indifféremment insérés à l'intérieur des parenthèses.
2. Les parenthèses peuvent être suivies par ?, +, ou \*.
3. Imbrication d'éléments mis entre parenthèses

## Exemple

```
<!ELEMENT cercle (centre, (radian|rayon))>
```

```
<!ELEMENT nom (nom_famille | (prenom, ((second_prenom+, nom_famille)  
| (nom_famille?)))>
```

# Modèle Mixte

**Spécifie un contenu comprenant des données textuelles et des éléments fils**

- **Notation**

`<!ELEMENT nom_type (#PCDATA| ...| nom_fils_n)*>`

- **Exemple**

```
<?xml version= "1.1" encoding= "ISO-8859-1" standalone= "yes" ?>
<!DOCTYPE monMessage [
  <!ELEMENT monMessage (#PCDATA| message)*>
  <!ELEMENT message (#PCDATA)>]>
<monMessage> ceci est un texte qui précède les deux éléments messages suivants
<message> considérez l'exercice sur la bibliographie de Alan Turing </message>
<message> définir la DTD associée </message> </monMessage>
```

## Exemple DTD bibliographie

```
<!DOCTYPE bibliographie [  
  <!ELEMENT bibliographie (#PCDATA | nom | profession |emphase | definition  
    | date)*>  
  <!ELEMENT definition (#PCDATA| terme)*>  
  <!ELEMENT terme (#PCDATA)>  
  <!ELEMENT profession (#PCDATA)>  
  <!ELEMENT emphase (#PCDATA)>  
  <!ELEMENT nom ((prenom, nom_famille)|nom_famille)>  
  <!ELEMENT date (jour, mois, an)>  
  <!ELEMENT jour (#PCDATA)>  
  <!ELEMENT mois ((#PCDATA)>  
  <!ELEMENT an (#PCDATA)>]>
```

## Élément vide

**Spécifie un élément dont le contenu est obligatoirement vide**

- **Notation**

`<!ELEMENT nom_type EMPTY>`

- **Propriété**

Ne peut être combiné avec un autre modèle

### Exemples

`<!ELEMENT image EMPTY>`

`<!ELEMENT erreur1 (foo |EMPTY)> <!--ERREUR-->`

`<!ELEMENT erreur2 (#PCDATA |EMPTY)> <!--ERREUR-->`

`<!ELEMENT exemple (foo|bar)>`

`<!ELEMENT foo (#PCDATA)>`

`<!ELEMENT bar EMPTY>`



## Elément potentiellement vide

**spécifié par un modèle #PCDATA**

### Exemple

```
<?xml version= "1.1" encoding= "ISO-8859-1" standalone= "yes" ?>
```

```
<!DOCTYPE exemple [
```

```
<!ELEMENT exemple (contenu)>
```

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

```
]>
```

```
<exemple><contenu> </contenu> </exemple>
```

ou

```
<exemple><contenu> peut également contenir du texte </contenu> </exemple>
```

## Exemples d'éléments

<!ELEMENT class (number, (instructor|assistant+),(credit|noCredit))>

<!ELEMENT donutBox (jelly?, lemon\*,((creme|sugar)+|glazed))>

<!ELEMENT farm  
 (farmer+,(dog\*|cat?),pig\*,(goat|cow)?,(chicken+|duck\*))>

Donner différent contenu conformes aux éléments

## Déclaration d'attribut

**Spécifie tous les attributs associés à un type d'éléments**

- **Notation**

`<!ATTLIST nomType nomAttribut typeAttribut déclarationDéfaut>`

- **Propriétés**

1. un type d'élément peut avoir un nombre arbitraire d'attributs
2. un attribut commun à plusieurs type doit être déclaré pour chaque type

### **Exemple**

```
<!ATTLIST image source CDATA #REQUIRED>
```

## Exemple: Attributs du type d'élément *image*

**Une simple instruction ATTLIST peut déclarer différents attributs pour le même document**

- **Notation**

```
<!ATTLIST nomType nomAttribut1 typeAttribut1 déclarationDéfaut1  
          nomAttribut_n typeAttribut_n déclarationDéfaut_n >
```

- **Exemple**

```
<!ATTLIST      image      source CDATA #REQUIRED  
                width CDATA #REQUIRED  
                height CDATA #REQUIRED  
                alt NMTOKENS #IMPLIED>
```

```
<image source="bus.jpg" width="152" height="345"  
      alt="nuage noir après passage de Gaston en voiture"/>
```

## Déclaration par défaut

**Spécifie si l'attribut est obligatoire ou la valeur par défaut de l'attribut**

**#IMPLIED** → attribut optionnel, aucune valeur par défaut n'est fournie

**#REQUIRED** → attribut obligatoire

**#FIXED 'value'** → valeur de l'attribut est fixe et non modifiable

**littérale** → vraie valeur par défaut en tant que chaîne entre guillemets

### Exemples

```
<!ELEMENT rectangle EMPTY>  
<!ATTLIST rectangle largeur CDATA "0">  
<rectangle largeur="100"/>
```

```
<!ATTLIST contact fax CDATA #IMPLIED>  
<contact fax="555-667788"/> ou <contact />
```

## Déclaration par défaut: Exemple valeur défaut

```
<!ATTLIST employeur compagnie CDATA #FIXED "Dauphine">  
<employeur compagnie="Dauphine"/>
```

```
<!--ERREUR-->  
< employeur compagnie="Paris VI"/>
```

```
<!ATTLIST biographie xmlns:xlink CDATA #fixed  
    "http://www.w3.org/1999:xlink">
```

```
<!ATTLIST page_web protocole NMTOKEN "http">
```

## Types d'attributs

**Spécifie des contraintes sur la valeur des attributs:unicité, choix dans une liste de valeurs établies, etc...**

- CDATA
- NMTOKEN, NMTOKENS
- ENUMERATION
- ID
- IDREF
- IDREFS
- ENTITY
- ENTITIES
- NOTATION

## Type CDATA

**Spécifie que la valeur de l'attribut est textuelle**

- **Propriété**

Utilisé pour les URI, adresses email, citations, toutes formes textuelles ne répondant pas autres types plus contraignants

- **Exemple**

```
<!ATTLIST      rapport
                langue CDATA #REQUIRED
                date_modification CDATA # REQUIRED
                diffusion CDATA # IMPLIED>
```

```
<rapport langue='FR' date_modification='20-MAI-2002' diffusion= 'confidentiel' >
```



## Type NMTOKEN

**Spécifie que la valeur de l'attribut est une unité lexicale nominale xml**

- Unité lexicale nominale  $\approx$  nom XML  
→ alphanumérique et des signes de ponctuation `_`, `-`, `.` et :
- exemple

```
<!ATTLIST journal mois NMTOKEN #REQUIRED>
```

## Type NMTOKENS

**Spécifie que la valeur d'un attributs contient une ou plusieurs unités lexicales nominales XML séparées par des blancs**

### Exemple

```
<!ATTLIST concerts dates NMTOKENS #REQUIRED>
```

```
<concerts dates=" Mai-15-2013 Oct-16-2013 Nov-15-2013">  
Orchestre de Radio France- salle Pleyel  
</concerts>
```

# Enumération

**Spécifie une liste de valeurs que peut prendre l'attribut dans un document**

- **Notation**

`<!ATTLIST nomElement nomAttribut (val1|...|valn) 'valDefault'>`

- **Exemple**

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

```
<ATTLIST date format (ANSI|ISO|EN-exp|FR-exp) #REQUIRED>
```

```
<date format="FR-exp"> 29 Mai 2004</date>
```

```
<date format="EN-exp"> 2004-05-29 </date>
```

## Exemple Enumération : Attributs de *Date*

```
<!ATTLIST date mois (janvier | février | Mars | Avril | Mai | Juin | Juillet | Août |  
    Septembre | Octobre | Novembre | Décembre) #REQUIRED>  
<!ATTLIST date jour (1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |  
    18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31) #REQUIRED>  
<!ATTLIST date an (1981 | 1982 | 1983 | 1984 | 1985 | 1986 | 1987 | 1988 | 1989 |  
    1990 | 1991 | 1992 | 1993 | 1994 | 1995 | 1996 | 1997 | 1998 | 1999 | 2000 | 2001  
    2003 | 2004 | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 30 | 31) #REQUIRED>  
<!ELEMENT date EMPTY>
```

```
<date mois="janvier" jour="22" an="2001"/>  
<date mois="janv" jour="22" an="2001"/>  
<date mois="01" jour="22" an="2001"/>  
<date mois="janvier" jour="02" an="2001"/>  
<date mois="janvier" jour="22" an="1980"/>
```

## Type ID

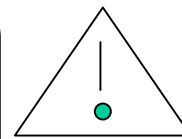
**Spécifie que la valeur doit être un nom XML qui est unique dans le document**

- **Propriétés**

1. Aucun autre attribut de type ID dans le document ne peut avoir la même valeur
2. Les attributs qui ne sont pas de type ID ne sont pris en compte
3. Chaque élément ne peut avoir qu'un seul attribut de type ID

- **Exemple**

```
<!ELEMENT personne (nom, action*)>
<!ATTLIST personne num ID #REQUIRED>
<personne num="SS123-45-6789">
  <nom> <famille> Lagaffe </famille> </nom>
</personne>
```



un nombre n'est  
pas un nom XML

## Type IDREF

**Spécifie que l'attribut fait référence à un attribut de type ID d'un élément du document**

- **Propriétés**

1. cet attribut doit être un nom XML
2. exprime les relations M-N ou 1-N

- **Exemples**

```
<!--fichier personne.dtd-->
<!ELEMENT personne (nom, action*)>
<!ATTLIST personne num ID #REQUIRED>
< ELEMENT action EMPTY>
<!ATTLIST action projet IDREF #REQUIRED>
<!ELEMENT nom ((prenom, nom_famille)
               | nom_famille)>
```

```
<!ELEMENT projet (but, membre+)>
<!ELEMENT but (#PCDATA)>
<!ELEMENT membre EMPTY>
<!ATTLIST projet num ID #REQUIRED>
<!ATTLIST membre personne IDREF
#REQUIRED>
```

## Exemple de document

```
<?xml version= "1.1" encoding= "ISO-8859-1" standalone= "no" ?>
```

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

```
<projet num= "p1">
```

```
<but>
```

```
Plan de développement stratégique
```

```
</but>
```

```
<membre personne= "ss123-45-6789"/>
```

```
<membre personne= "ss879-64-4587"/>
```

```
</projet>
```

```
<projet num= "p2">
```

```
<but> Apprentissage de XML</but>
```

```
<membre personne= "ss765-45-6789"/>
```

```
<membre personne= "ss879-64-4587"/>
```

```
</projet>
```

```
<personne num= " ss123-45-6789 ">
```

```
<nom> <famille> Lagaffe </famille> </nom>
```

```
<action projet = "p1"/>
```

```
</personne>
```

```
<personne num=" ss879-64-4587 ">
```

```
<nom> <famille> Montes </famille> </nom>
```

```
<action projet = "p1"/>
```

```
<action projet = "p2"/>
```

```
</personne>
```

```
<personne num=" ss765-45-6789 ">
```

```
<nom> <famille> Dupont </famille> </nom>
```

```
<action projet = "p2"/>
```

```
</personne>
```

## Type IDREFS

**Spécifie que l'attribut contient une liste de noms XML séparés par des blancs, chacun d'eux devront être l'ID d'un élément du document**

### Propriété

Utilisé lorsqu'un élément fait référence à plusieurs éléments de même type

### Exemples

```
<!ATTLIST action projet IDREFS #REQUIRED>
<!ATTLIST membre personne IDREFS #REQUIRED>
<projet num="p2">
<but> Apprentissage de XML</but>
<membre personne="ss765-45-6789 ss879-64-4587"/>
</projet>
```

```
<personne num=" ss879-64-4587 ">
<nom>
<famille> Montes </famille>
</nom>
<action projet ="p1 p2"/>
</personne>
```



## Déclaration d'entités générales

**Associe un nom d'entité à un texte de substitution xml bien formé.**

- Notation

`<!ENTITY nomEntite "texte de substitution">`

- Propriétés

Le nom de l'entité doit être un nom XML

- Exemples

`<!ENTITY chaine_longue "chaîne de caractères trop longue">`

`<!ENTITY super "supercalifragilisticexpealidocious">`

## Référence à une entité générale

**Toute entité générale déclarée dans une DTD peut être référencée dans la DTD et dans le contenu du document**

- **Notation**

&nomEntite;

- **Propriété**

la référence d'une entité est indépendante de l'ordre des déclarations

- **Exemple**

```
<!Element test (contenu, edition)>
<!ENTITY lab "&abrev; &long;">
<!ENTITY abrev "ceci est une phrase">
<!ENTITY long "très très longue">
<!ENTITY cp "&#xA9; Edition La Lune Blanche">
```

```
<test>
<contenu>
&lab; &amp; un peu obscure
</contenu>
<edition> &cp; </edition>
</test>
```

## Déclaration d'entité générale externe

**Entité dont le texte associé est défini dans un fichier externe à la DTD du document**

- **Notation**

`<!ENTITY nomEntite SYSTEM "URI_ou_URL_relative">`

- **Exemple**

`<!ELEMENT brevet (titre, auteur, chapitre*)>`

`<!ENTITY chapitre_1 SYSTEM "../chapitre1.xml">`

`<!ENTITY chapitre_2 SYSTEM "../chapitre2.xml">`

`<!ENTITY gl "Gaston Lagaffe">`

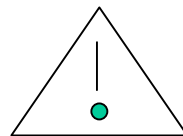
`< brevet > <titre> Manuel de référence de la machine infernale </titre>`

`<auteur> &gl; </auteur>`

`&chapitre_1;`

`&chapitre_2;`

`</ brevet >`



chapitre1 et chapitre2  
doivent être des documents xml  
bien formés

## Déclaration de texte

**Introduit le texte bien formé xml associé à une entité externe analysée**

- **Notation**

```
<?xml version="valeur" encoding="MacRoman"?>
```

```
<?xml encoding="MacRoman"?>
```

- **Propriété**

La déclaration d'encodage est obligatoire

- **Exemple**

```
<!--chapitre1.xml-->
```

```
<?xml encoding="ISO-8859-1"?>
```

```
<chapitre titre ="chapitre 1">
```

La machine infernale est le fruit de l'imagination d'un doux rêveur ....

```
</chapitre>
```

## Déclaration d'entités paramètres

**Spécifie une entité associé à un texte analysable uniquement dans une DTD.**

- **Notation**

`<!ENTITY % entite_paramètre "texte associé">`

Attention !!!

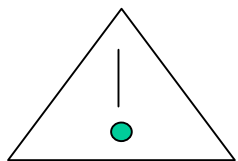
Espace entre % et le nom de l'entité paramètre

- **Propriétés**

1. Uniquement référencée dans une DTD
2. Ne contient que du XML bien formé
3. Permet d'utiliser plusieurs fois un modèle de contenu, une liste d'attributs identiques dans différentes déclarations d'éléments

## Référence d'entité paramètre

- Notation  
%entiteParametre;
  - Propriétés
    1. Une entité générale peut référencée une entité paramètre
    2. Une entité paramètre peut référencée une entité générale
- => Le remplacement des références ne s'effecute que lors du traitement du document xml



Une valeur  
d'attribut ne peut  
référencer une  
entité paramètre

```
<!--ERREUR -->  
<!ENTITY % n1 "numéroté">  
<!ENTITY % n2 "ordonné">  
<!ELEMENT liste (element+)>  
<!ELEMENT element (#PCDATA)>  
<!ATTLIST liste type (%n1; | %n2; ) #REQUIRED >
```

# Solution

```
<!DOCTYPE exemple [  
<!ENTITY % value_type "(numéroté| ordonné) #REQUIRED">  
<!ELEMENT exemple (liste+)>  
<!ELEMENT liste (element+)>  
<!ELEMENT element (#PCDATA)>  
<!ATTLIST liste type %value_type;>]>  
<exemple>  
  <liste type="ordonné">  
    <element> ceci est un test</element>  
  </liste>  
  <liste type="classé">  
    <element> ceci est un test</element>  
  </liste>  
</exemple>
```

## ERREUR

L'attribut "type" de valeur  
"classé" doit avoir une valeur issue  
de la liste "numéroté ordonné ". [38]

## Exemple Référence d'entités paramètres livre.dtd

```
<!ENTITY % contenu "titre, auteur, chapitre*">
<!ELEMENT livre (% contenu ; , reference)>
<!ELEMENT reference (#PCDATA)>
<!ENTITY % debut "titre, intro">
<!ELEMENT chapitre (% debut; , section+)>
<!ELEMENT titre (#PCDATA)>
<!ELEMENT intro (#PCDATA)>
<!ELEMENT section (% debut; , sous-section+)>
<!ELEMENT sous-section (% debut; , sous-sous-section+)>
<!ELEMENT sous-sous-section (% debut; , paragraphe+)>
<!ELEMENT paragraphe (#PCDATA)>
<!ENTITY % publication "&#XC9; dition Eyrolles">
<!ENTITY rights "tous droits réservés">
<!ENTITY titre "La Machine Infernale">
<!ENTITY reference "&gl; &titre; &#xA9; 1970 %publication; &rights;">
```

```
<livre>
<titre> &titre; </titre>
<auteur> &gl; </auteur>
&chapitre_1;
&chapitre_2;
<reference>
&reference;
</reference>
</livre>
```



## Redéfinition des entités paramètres

**Une entité paramètre définie dans un fichier externe peut être redéfinie dans la DTD interne d'un document**

- **propriété**

La déclaration d'entité paramètre traitée en premier est prioritaire

⇒ La redéfinition d'une entité paramètre redéfinie dans une DTD interne est prioritaire

- **exemple**

```
<!DOCTYPE livre_universitaire SYSTEM livre.dtd [  
<!ENTITY % contenu " titre, sous_titre?, auteur, chapitre*, exercice+">]
```

## Exemple suite

```
<?xml version= "1.1 " standalone= "no">
<!DOCTYPE livre_universitaire SYSTEM livre.dtd [
<!ENTITY % contenu " titre, sous_titre?, auteur, (chapitre, exercice*)+">
<!ENTITY chapitre1 SYSTEM "../chapitre1.xml">
<!ELEMENT sous_titre (#PCDATA)>
<!ELEMENT exercice (#PCDATA)>
<!ATTLIST exercice titre CDATA #REQUIRED>]
<livre_universitaire> <titre> XML facile </titre>
    <sous_titre> Comment apprendre XML en un jour </sous_titre>
    <auteur> M.J. Bellosta </auteur>
    &chapitre1;
</exercice titre = "Exercice 1">
Définissez une déclaration xml de version 1.1 et sans dtd externe.</exercice>
<reference>
xml in a nutshell, Manuel de référence, traduction Thomas Broyer,eds O'Reilly
</reference>
</livre_universitaire>
```

## Entité paramètre externe

**Associe une DTD partielle définie dans un fichier à une entité paramètre**

- **Notation**

```
<!ENTITY % entite_paramètre SYSTEM "URI_ou_URL">
```

- **Propriétés**

Permet de définir une DTD en plusieurs DTDs, toutes associées à des entités paramètres externes.

- **Exemple**

```
<!ENTITY % names SYSTEM "./names.dtd">  
<!ENTITY % personne SYSTEM "./personne.dtd">  
<!ENTITY % livre SYSTEM "./livre.dtd">
```