

Structuration et échange d'informations sur le Web

Université Lyon 1
Master CCI

L. Médini, Janvier 2006

Plan du cours 1 : Principes fondamentaux, syntaxe et validation

- Introduction
 - Définitions
 - Aperçu de quelques langages documentaires
 - Les langages dédiés au web
- HTML, XML et XHTML
 - Principes de base
 - Syntaxe
- Validation de documents XML : les DTD
- Feuilles de style CSS

Plan du cours 2 : Applications et programmation XML

□ Applications XML

- Notion d'espaces de noms XML
- Retour sur la validation de documents : les schémas XML
- Le langage de feuilles de style XSL
 - Xpath
 - XSLT

□ Programmation XML

- Les API existantes
- Le Document Object Model (DOM)
- Simple API for XML (SAX)

Introduction

- De quoi on va parler dans ce cours ?
 - De « Nouvelles Technologies de l'Information et de la Communication »
 - Langages dédiés au web (XHTML, XML...)
 - Programmation web
- Points de vue abordés
 - Structuration des données
 - Échange et partage des données
- Ce qu'on ne fera pas
 - Applications web (« services Web »)
 - Aspects sémantiques (« Web sémantique »)

Introduction

□ Définitions

- L'information en tant que signal circulant entre un émetteur et un récepteur lors d'un processus de communication
 - Origine : théorie de l'information (Shannon 1948), traitement du signal (Fourier ~1800)
 - On s'intéresse uniquement à l'aspect physique du signal transmis
 - Émetteur et récepteur peuvent être humains ou logiciels.

Introduction

□ Définitions

- L'information en tant que message véhiculé lors d'un processus de communication
 - On s'intéresse au contenu de l'information
 - Émetteur et récepteur logiciels : traitement de l'information, informatique
 - Émetteur et récepteur humains : linguistique, sémiologie (De Saussure 1916)
 - Extension de cette notion par la prise en compte du contexte général de la communication : sémiotique (Pierce ~1870)

Introduction

□ Définitions

- Donnée : valeur associée à un type de données
 - Décrite par des caractéristiques de forme
 - Indépendante de son interprétation : donnée « brute »
- Élément d'information : ensemble de données faisant « sens »
 - On parle aussi de « grains d'information »
- Document : regroupement cohérent d'éléments d'information
 - Autour d'une thématique commune
 - Selon une structure donnée

Introduction

□ Définitions

- Information structurée : bases de données
 - Éléments d'information stockés séparément
 - Accès par requêtes
 - Traitements facilités
- Information non structurée : corpus documentaires
 - Éléments d'information stockés sous forme de textes
 - Accès par recherche d'information
 - Traitements complexes
- Information semi-structurée : langages à balises
 - Éléments d'information stockés dans des documents
 - Permet les deux types d'accès et de traitements

Introduction

□ Définitions

- Balise : signe marquant une position particulière
 - Signe = élément d'information
 - Marquer = permettre la distinction
 - Ex : fusée de détresse, signal radio, « tag » HTML
 - Position = par rapport à l'espace considéré
 - Ex : espace 3D (avion), 2D (bateau), 1D (document)
 - Particulière = en général, la position d'un élément de l'espace qu'on cherche à repérer
 - Ex : aéroport, bateau, élément d'information

⇒ Type d'information facilement repérable permettant d'identifier d'autres éléments informationnels (i.e. « méta-information »)

Introduction

□ Définitions

■ Balisage documentaire : utilisation de balises

- Pour marquer des points précis d'un document
- Pour marquer des zones (segments) de document
 - Balisage de début et de fin de zone
- Pour structurer le document

⇒ Utilisation de plusieurs types de balises

- ⇒ En fonction du type d'élément à marquer
- ⇒ En fonction du type de marquage (point, début, fin)

■ Langage à balises : langage de description de documents utilisant ces techniques de balisage

■ Élément : une balise et son contenu

Introduction

□ Quelques langages à balises

- Langages de *balisage procédural* : permettent de décrire la mise en forme (formatage) d'un document
 - Ex : PS, RTF, TeX, HTML
- Langages de *balisage descriptif* : se contentent de décrire les données, sans but de traitement
 - Ex : RDF, OWL, MathML
- *Méta-langages* de balisage : permettent de définir des langages de balisage
 - Ex : SGML, XML

Introduction

□ Le langage Postscript

```
%!PS-Adobe-3.0
%%Title: Microsoft Word -
Document1
%%Creator: PSCRIPT.DRV
Version 4.0
%%CreationDate: 03/02/02
10:47:00
...
%%EndComments
%%BeginProlog
%%BeginProcSet:
Pscript_Res_Emul 1.0 0
/defineresource
where{ pop}{ userdict
begin/defineresource{ userdi
ct/Resources 2
...
}ifelse}bind readonly def
end}ifelse
%%EndProcSet
```

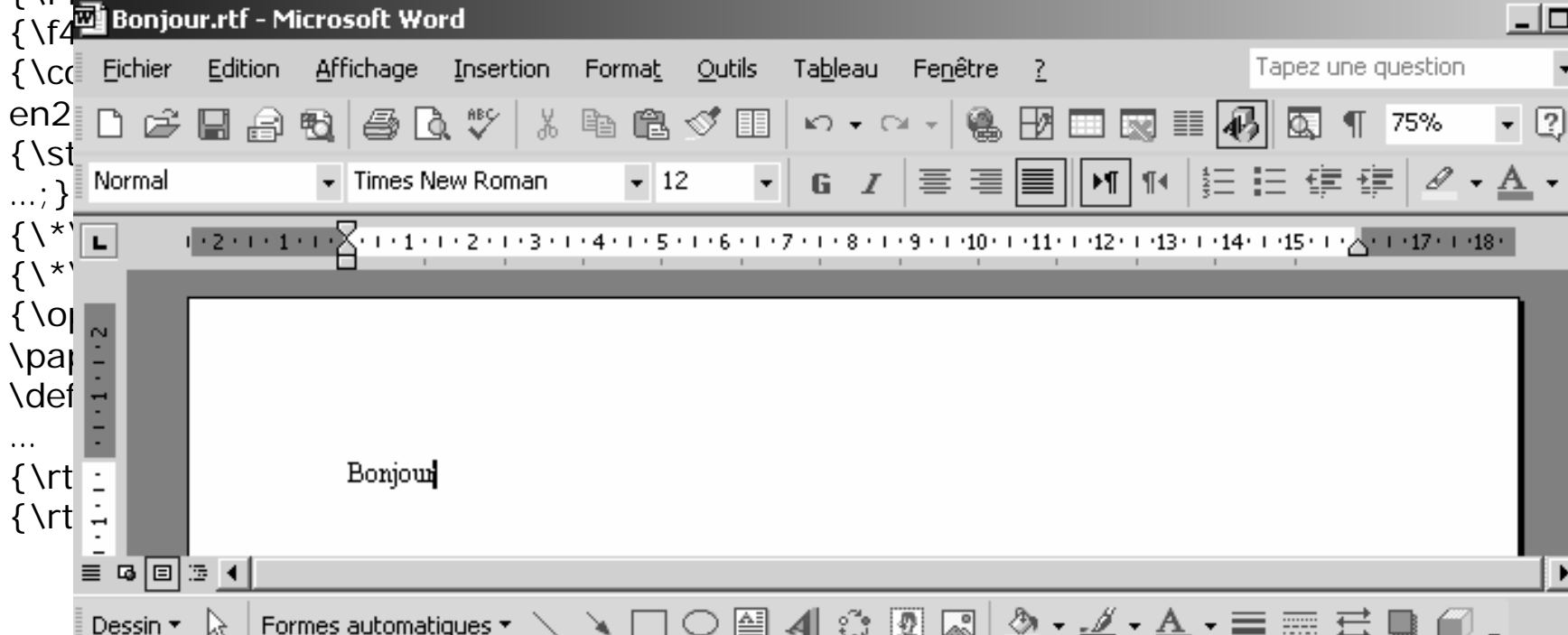
```
%%BeginResource: file
...
%%EndResource
...
%%EndProlog
%%BeginSetup
...
%%BeginFeature:
*PageSize A4
...
%%EndFeature
...
%%EndSetup
%%Page: 1 1
%%BeginPageSetup
...
%%EndPageSetup
...
```

```
%%IncludeFont: Courier
...
(Courier) cvn /Type1
...
(Essai impression)S
...
(%%[ Page: 1 ]%%) =
%%PageTrailer
%%Trailer
%%DocumentNeededFonts:
%%DocumentSuppliedFonts:
/Pscript_Win_Driver /ProcSet
findresource dup /terminate g
exec
Pscript_Win_Compat dup
/terminate get exec
%%Pages: 1
(%%[ LastPage ]%%) =
%%EOF
```

Introduction

▣ Le langage RTF

```
{\rtf1\adeflang1025\ansi\ansicpg1252\uc1\adef0\deff0\stshfdbch0\stshfloch0\stshfhich0\stshfbi0\deflang1036\deflangfe1036{\fonttbl{\f0\froman\fcharset0\prq2{\*\panose 02020603050405020304}Times New Roman;}{\f36\froman\fcharset238\prq2 Times New Roman CE;}{\f37\froman\fcharset204\prq2 Times New Roman Cyr; }...{\f42\froman\fcharset186\prq2 Times New Roman Baltic;}
```



Introduction

□ Les langages dédiés au Web

■ Historique 1/3

- 1960-1986 : SGML (norme ISO)
- 1989 : ODA (norme ISO, concurrent de SGML)
- Fin des années 80 : apparition/essor du web
- 1992-1997 : HTML (versions 1.0 -> 4.01)
- Octobre 1994 : création du World Wide Web Consortium (W3C) : <http://www.w3.org>
- 1996-1999 : CSS Level 1 (fonctionnalités de base)
- Février 1998 : XML version 1.0
- 1998 : CSS Level 2 (fonctionnalités supplémentaires)
- Octobre 1998 DOM Level 1 (supporte XML et HTML)

Introduction

□ Les langages dédiés au Web

■ Historique 2/3

- Décembre 1999 : XHTML 1.0
- 1999-2004 : RDF et RDF-Schema 1.0
- Novembre 2000 : DOM Level 2 (supporte CSS et espace de noms XML)
- Mai 2001 : schémas XML 1.0
- Juin 2001 : XLink 1.0
- Juillet 2001 : SVG 1.0
- Novembre 2001 : XSL 1.0
- Janvier 2003 : SVG 1.1

Introduction

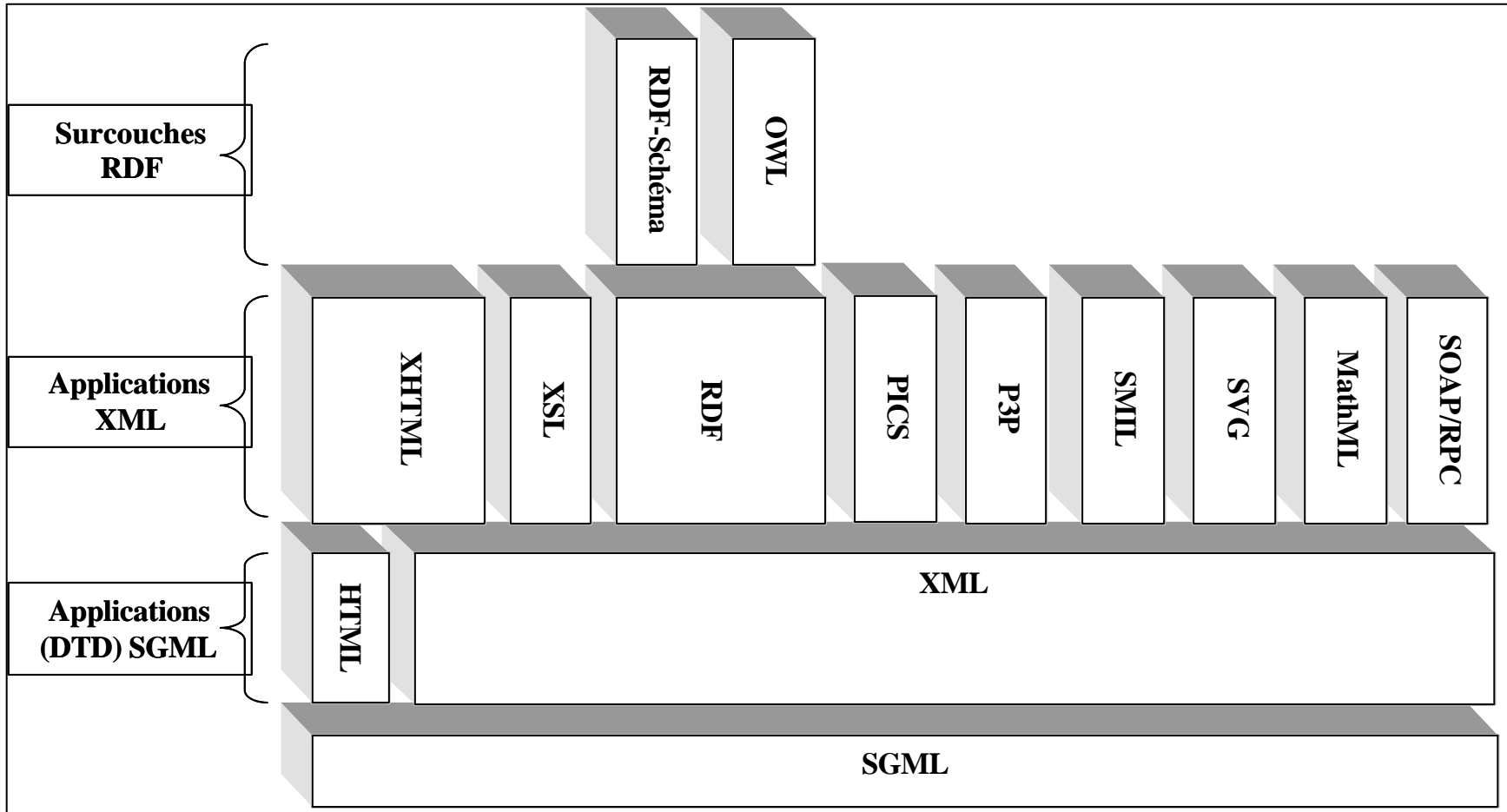
□ Les langages dédiés au Web

■ Historique 3/3

- Février 2004 : XML 1.1
- Février 2004 : OWL 1.0
- Avril 2004 : DOM Level 3 Core
- Octobre 2004 : XML Schema (2^e édition)
- Octobre 2004 : XQuery 1.0
- Octobre 2004 : XPath 2.0 (Working Draft)
- Novembre 2004 : XSLT 2.0 (Working Draft)
- Décembre 2004 : WSDL 2.0
- Restent en développement : CSS L3, DOM L3...

Introduction

▣ Quelques langages dédiés au Web



HTML, XML et XHTML

□ HTML

- DTD SGML : ensemble (dé)fini d'éléments SGML
- Destiné à visualiser des hyperdocuments sur écran
- Interprété côté client par les navigateurs
- V. 1.0 : 1992 -> V. 4.01 (dernière) : oct. 1997
- Souvent associé à d'autres langages
 - CSS : langage de feuilles de style
 - Langages de scripts : JavaScript, JScript, VBscript...
 - Inclusion d'objets définis dans d'autres langages : applets Java, contrôles ActiveX...
- Remarque : le langage D(H)TML n'existe pas

HTML, XML et XHTML

□ XML : principes de base

- DTD de SGML (beaucoup plus concis)
- Restrictions de syntaxe et non de contenu
- Au même titre que SGML, c'est un méta-langage de description des données
 - ⇒ Ça ne sert à rien
 - ⇒ Ça permet de définir des « applications » pour faire ce qu'on veut avec
 - Visualiser des pages web
 - Décrire des images
 - Échanger des données techniques...
- V 1.0 : fév. 1998 -> 04 fév. 2004 (3ème édition)
- V 1.1 : 04 février 2004

HTML, XML et XHTML

□ XML : composants

- XML (syntaxe) : documents « bien formés »
- DTD/schémas XML : documents « valides »
- Processeur (parser) XML : analyse et traitement
- DOM : modèle arborescent des données d'un document pour un langage de programmation
- SAX : accès « simple » aux données (programmation événementielle)
- Espaces de noms (« namespaces ») : interopérabilité des applications
- XBase, XPointer, XLink : mécanismes de liens
- XSL : mécanismes de transformation

HTML, XML et XHTML

- Syntaxe XML (voir poly p. 50) : un document XML bien formé est composé

- D'un prologue, contenant :

- Éventuellement une déclaration XML

```
<?xml version="1.0" encoding="ISO-8859-1"
standalone="yes"?>
```

- Éventuellement une déclaration de type de document

```
<!DOCTYPE Nom_de_l_élément_racine Type_de_source
emplacement1 emplacement2 [sous-ensemble interne
de DTD]>
```

- D'un élément « racine », composé :

- Soit d'une balise ouvrante, éventuellement d'un contenu et d'une balise fermante

```
<NomElement attribut1="valeur1" attribut2="valeur2"...>
contenu
</NomElement>
```

HTML, XML et XHTML

- Syntaxe XML (voir poly p. 50) : un document XML bien formé est composé
 - D'un prologue, contenant :
 - Éventuellement une déclaration XML

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
```
 - Éventuellement une déclaration de type de document

```
<!DOCTYPE Nom_de_l_élément_racine Type_de_source emplacement1 emplacement2 [sous-ensemble interne de DTD]>
```
 - D'un élément « racine », composé :
 - Soit d'une balise d'élément vide

```
<Nom_element_vide att1="val1" att2="val2"... />
```

HTML, XML et XHTML

□ Syntaxe XML (voir poly p. 50) : remarques

■ Structure

- Un contenu est composé de texte et / ou d'autres éléments appelés fils de l'élément courant
- Le texte est appelé « données caractères analysables » ou PCDATA (pour Parsed Character DATA).
- Le contenu autorisé pour le PCDATA dépend du type d'encodage choisi
- Le fait d'avoir un unique élément racine, des éléments fils, eux-mêmes décomposables, etc. définit une **structure arborescente**
- Pas de chevauchement de balises entre un élément père et un élément fils

■ Caractères spéciaux

- Les caractères "<" (inférieur) et "&" (esperluète) sont interdits dans les contenus. On aura recours aux entités "<" et "&".
- L'usage de ">" (supérieur) ou des guillemets simples ou doubles peut également être perturbant. Dans ce cas, on a recours à ">", "'" et """.
- Si l'on veut vraiment utiliser les caractères "<" ou "&", il est possible de définir une balise sous forme de zone de caractères non analysés (CDATA), sous la forme :

```
<![CDATA [ texte comprenant des caractères interdits ]]>
```

HTML, XML et XHTML

□ XHTML

- DTD XML : ensemble (dé)fini de balises XML
- But : réécrire HTML, mais en plus propre
- V. 1.0 : janv. 2000, révisée en 2002
- V. 1.1 : mai 2001 (pas de rapport avec XML 1.1)
- Trois recommandations
 - XHTML frameset
 - XHTML transitionnel
 - XHTML strict

⇒ Séparation du fond et de la forme

⇒ Pris en charge par tous les navigateurs récents

⇒ Pris en charge par les outils de traitement XML

HTML, XML et XHTML

- ▣ Syntaxe (X)HTML : poly page 21

Document XML valide : les DTD

□ Généralités

- Héritées de SGML
- Syntaxe : grammaire EBNF
- Partie intégrante de la recommandation XML
- Reconnues par tous les parsers XML validants

□ Composants

- Déclaration de type de document (dans le document XML)
- Définition de type de document (interne ou externe)

Document XML valide : les DTD

- Déclaration de type de document
 - Une et une seule par document XML
 - Placée
 - Après la déclaration XML
 - Avant le début de l'élément racine du document
 - Syntaxe générique

```
<!DOCTYPE Nom_élément_racine Type_de_source  
emplacement1 emplacement2 [sous-ensemble  
interne de DTD]>
```

Document XML valide : les DTD

□ Déclaration de type de document

```
<!DOCTYPE Nom_élément_racine Type_de_source emplacement1  
    emplacement2 [sous-ensemble interne de DTD]>
```

■ Types de sources

- Emplacements des fichiers DTD

■ Sources de type SYSTEM

- DTD internes

```
<!DOCTYPE Nom_élément_racine SYSTEM [sous-ensemble interne]>
```

- DTD externes

- Un seul emplacement possible

- Indiqué par une URI

```
<!DOCTYPE Nom_élément_racine SYSTEM "../DTD/nom_fichier.dtd">
```

```
<!DOCTYPE Nom_élément_racine SYSTEM
```

```
"http://www.une_url_quelconque.com.nom_fichier.dtd">
```

Document XML valide : les DTD

□ Déclaration de type de document

```
<!DOCTYPE Nom_élément_racine Type_de_source emplacement1  
    emplacement2 [sous-ensemble interne de DTD]>
```

■ Sources de type PUBLIC

- Uniquement des DTD externes
- Un ou deux emplacements possibles
- Indiqués par

- Un identificateur public (reconnu par l'application)

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
```

- Un identificateur public + une URI

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

Document XML valide : les DTD

□ Définition de type de document

■ Quatre types de données

□ ELEMENT : un élément XML et son contenu

- Syntaxe : `<!ELEMENT nom_elt (contenu)>`
- Types de contenu
 - Autres élément(s)
 - #PCDATA (pour « Parsed Character DATA »)
 - mixte (élément(s) + PCDATA)
 - #ANY (n'importe quel type de contenu XML bien formé)
 - #EMPTY
- Pour éléments ou mixte : indiquer leur(s) nom(s) séparés par des virgules (« , » : ET) ou des pipes (« | » : OU)
- Pour éléments, PCDATA ou mixte : contenus entre parenthèses

Document XML valide : les DTD

□ Définition de type de document

■ Quatre types de données

□ ELEMENT : un élément XML et son contenu

▪ Opérateurs de cardinalité

- "" (rien) : une et une seule occurrence
- "?" : zéro ou une occurrence
- "*" : zéro, une ou plusieurs occurrences
- "+" : une ou plusieurs occurrences

Remarque : les opérateurs de cardinalité sont postfixés

- Exemple de définition d'élément

Document XML valide : les DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Texte SYSTEM "Texte.dtd">
<Texte>
  <Titre>
    <Mot>XML</Mot>
  </Titre>
  <Phrase>
    <Mot>XML</Mot>
    <Mot>c</Mot>
    <Ponctuation>'</Ponctuation>
    <Mot>est</Mot>
    <Mot>chouette</Mot>
    <Ponctuation>.</Ponctuation>
  </Phrase>
</Texte>
```

```
<!ELEMENT Texte (Titre, Phrase+)>
<!ELEMENT Titre (Mot+, (Ponctuation, Mot+)*)>
<!ELEMENT Phrase (Ponctuation?, (Mot+, Ponctuation)+)>
<!ELEMENT Mot (#PCDATA)>
<!ELEMENT Ponctuation (#PCDATA)>
```

Document XML valide : les DTD

□ Définition de type de document

■ Quatre types de données

□ ATTLIST : ensemble des attributs d'un élément XML

▪ Syntaxe

```
<!ATTLIST nom_element  
  nom_att type_att déclaration_valeur_implicit >
```

▪ Types d'attributs

- CDATA : chaîne textuelle simple
- Valeurs énumérées dans une liste entre parenthèses et séparés par des pipes.
- ID : identifiant unique ; un seul attribut ID par élément
- IDREF : référence à un élément identifié par son attribut ID
- IDREFS : liste de plusieurs IDREF séparés par des espaces
- NMTOKEN : jeton de nom qui permet de limiter le nombre de valeurs que peut prendre l'attribut
- NMTOKENS : liste de jetons de noms séparés par des espaces
- ENTITY : nom d'une entité prédéfinie
- ENTITIES : liste de plusieurs entités séparés par des espaces
- NOTATION : type de notation déclaré ailleurs dans la DTD

Document XML valide : les DTD

□ Définition de type de document

■ Quatre types de données

□ ATTLIST : ensemble des attributs d'un élément XML

▪ Syntaxe

```
<!ATTLIST nom_element
```

```
  nom_att type_att déclaration_valeur_implicit>
```

▪ Déclaration de valeur implicite

- #REQUIRED : obligatoire
- #IMPLIED : facultatif sans valeur par défaut
- "valeur" : facultatif avec valeur par défaut
- #FIXED "valeur" : facultatif, mais s'il est présent, il doit obligatoirement avoir la valeur indiquée

Document XML valide : les DTD

```
<Livre titre="Développer en XML avec Java 2" auteurs="C. Daconta,
Al Saganich" isbn="2-7440-1099-5" editeur="CampusPress"
idlivre="L1">
  <Table_des_matières>
    ...
  </Table_des_matières>
  <Resume>
    ...
  </Resume>
</Livre>
```

```
<!ELEMENT Livre (Table_des_matières, Resume)>
```

```
<!ATTLIST Livre
```

titre	CDATA	#REQUIRED
isbn	CDATA	#REQUIRED
auteurs	CDATA	#REQUIRED
editeur	CDATA	#IMPLIED
date_publi	CDATA	#IMPLIED
prix	CDATA	"0"
avis	(TB B P)	#IMPLIED
idlivre	ID	#REQUIRED>

Document XML valide : les DTD

□ Définition de type de document

■ Quatre types de données

□ ENTITY : séquence de caractères repérée par un nom

▪ Syntaxes

```
<!ENTITY nom_entité "valeur">
```

```
<!ENTITY % nom_entité "valeur">
```

```
<!ENTITY nom_entité SYSTEM "URI">
```

```
<!ENTITY nom_entité PUBLIC "URI1" "URI2">
```

▪ Associent à nom_entité soit la valeur entre guillemets, soit la valeur définie dans la ressource pointée par les URI.

▪ Les deux premiers exemples définissent des « entités internes » et les deux derniers des « entités externes »

Document XML valide : les DTD

□ Définition de type de document

■ Quatre types de données

□ ENTITY : séquence de caractères repérée par un nom

▪ Syntaxes

```
<!ENTITY nom_entité "valeur">
```

```
<!ENTITY % nom_entité "valeur">
```

```
<!ENTITY nom_entité SYSTEM "URI">
```

```
<!ENTITY nom_entité PUBLIC "URI1" "URI2">
```

▪ Le deuxième exemple définit une « entité paramètre », qui n'est utilisable que dans la DTD et référencable par

`%nom_entité;`

▪ les trois autres définissent des « entités générales » référencables par `&nom_entité;` dans la DTD et dans le document XML

Document XML valide : les DTD

□ Définition de type de document

■ Quatre types de données

- NOTATION : séquence de caractères non XML destinées à être traitées par une application particulière

- Syntaxes

- ```
<!NOTATION nom_notation SYSTEM "URI">
```

- ```
<!NOTATION nom_notation PUBLIC "URI1" "URI2">
```

- Les notations sont déclarées comme des entités externes
- Il est possible d'utiliser les noms de notations comme types d'attributs

Feuilles de style HTML : CSS

- Buts : définir des caractéristiques de style
 - Standardisées pour être reconnues par tous les navigateurs
 - Dans un langage différent de HTML pour séparer la forme du fond
 - Externalisables pour pouvoir les réutiliser
 - Imbriquables pour pouvoir les appliquer à des sous-ensembles de documents

Feuilles de style HTML : CSS

□ Trois niveaux d'application

- Niveau élément : feuilles de style intégrées
 - Syntaxe : caractéristiques de style dans la valeur de l'attribut `style` de la balise ouvrante
- Niveau document : feuilles de style incorporées
 - Syntaxe : élément `<style type="text/css">` dans la balise `<head>` d'un document HTML
- Niveau site web : feuilles de style liées
 - Syntaxe
 - Caractéristiques de style dans un fichier à part
 - Lien vers ce fichier par son URI dans un document HTML :
`<link rel="stylesheet" type="text/css" href="URI" /`

Feuilles de style HTML : CSS

□ Syntaxe du langage

■ Définition des caractéristiques de style

```
nom_element {caract1:valeur1; caract2: valeur2;...}
```

```
.nom_classe {caract1:valeur1; caract2: valeur2;...}
```

```
nom_element.nom_classe {caract1:valeur1;...}
```

■ Application de styles

□ Aux éléments HTML : pas besoin de demander l'application

□ À des classes de style : application par l'attribut `class="nom_classe"`

■ Détail des caractéristiques : poly p. 37.