TP2 : Introduction à XML

Règles de nommage des balises:

Les noms des balises sont libres avec certaines règles :

- Pas d'espaces, pas d'apostrophe, pas de /
- Premier caractère alphabétique ou _
- Noms sensibles aux majuscules-minuscules
- Noms composés avec le caractère - autorisés: exemple: ordre-achat

Exercice N°1 :

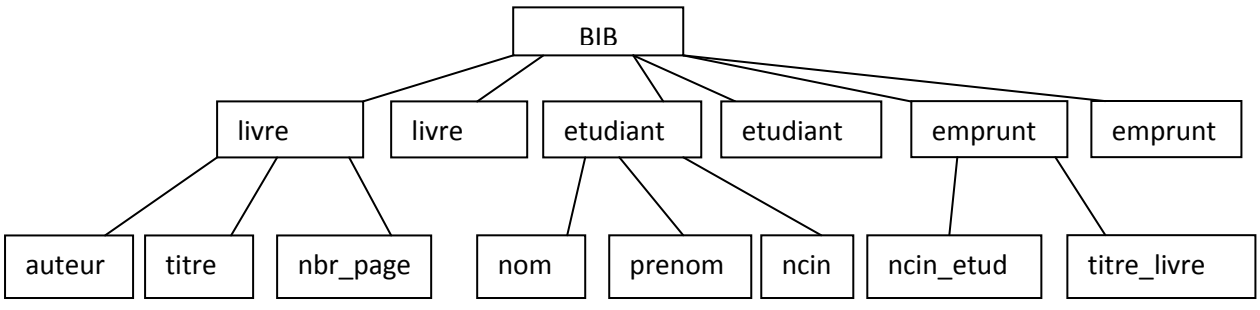
Identifier les balises XML correctes et celles erronées. Indiquer les erreurs.

1. <Drivers_License_Number>98 NY 32</Drivers_License_Number>
2. <Driver's_License_Number>98 NY 32</Driver's_License_Number>
3. <month-day-year>1/10/2005</month-day-year>
4. <first name>Alan</first name>
5. <àçttûä>øåú</àçttûä>
6. <PRENOM>Alan</prenom>
7. <month/day/year>1/10/2005</month/day/year>
8. <_4-lane>I-610</_4-lane>
9. <téléphone>0 123 456 789</telephone>
10. <4-lane>I-610</4-lane>

Exercice N°2 :

Le but de cet exercice est de vous initier à la création d'un document XML. Pour arriver à cette fin, vous allez procéder en étapes.

1. Créer un document XML.
2. Représenter le document selon une structure en arbre illustrée par la figure ci-dessous



Les données du document sont :

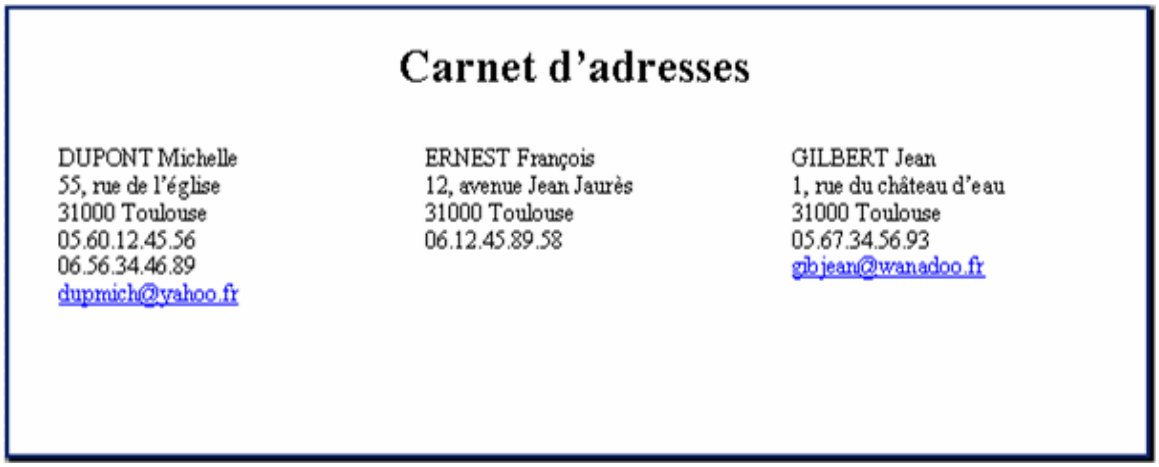
a. 3 Livres : (auteur1 ; titre1 ; 56), (auteur2 ; titre2 ; 100), (auteur3 ; titre3 ; 300).
b. 3 étudiants : (bensassi ; tasnim ; 1234567) ; (mhafdhi ; neila ; 1478523)
(bensassi ; ayoub ; 1258963).
c. 1 emprunt : (1258963 ; titre3).

3. Vérifiez que le document est bien formé.

Remarque : Un document XML bien formé (well formed) est un document XML syntaxiquement correct

Exercice N°3 :

Soit le carnet d'adresses suivant :



1. Representer sous forme d'un texte balisé le carnet d'adresse ci-dessus.
2. Vérifier que le document est bien formé.

Exercice N°4 :

Une entreprise de vente de matériel informatique désire garder les informations de son stock dans un document XML. Vous disposez des informations suivantes

- Le stock contient plusieurs produits.
- Chaque produit identifié par un identifiant unique (idprod) est reconnu par sa marque, son modèle et son fournisseur.
- Chaque produit appartient à une catégorie donnée.
- Chaque catégorie identifiée par « idcat » est reconnue par son libellé. Le idcat doit nécessairement commencer par la lettre C suivie de 5 chiffres.

- Chaque fournisseur identifié par « idfour » est reconnu par sa raison sociale, son adresse et son téléphone fixe. L'adresse du fournisseur ne doit pas dépasser les 40 caractères.

1. Ecrire un exemple de document XML répondant aux besoins de cette agence

Exercice N°5 :

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.

Annexe

- Une DTD définit une classe de documents. Elle définit les noms des éléments, attributs et entités et leur type.
- Un document est valide s'il est bien formé et est valide par rapport à la DTD c'est à dire le document est un arbre qui peut être produit par la DTD.
- Une DTD peut-être spécifiée à l'intérieur du document ou par un lien :
    - **Déclarations internes :**

      ```
      <?xml version="1.0"?>
      <!DOCTYPE racine [
      <!--Toutes les déclarations pour la DTD de racine se trouve dans cette section -->
      ...<!ELEMENT racine ... >...
      ]>
      <racine>
          <!-- Ceci est une instance d'un document de type racine -->
      </racine>
      ```

    - **Déclarations externes :**

      ```
      <?xml version="1.0"?>
      <!DOCTYPE racine SYSTEM "racine.dtd">
      <racine>
          <!-- instance d'un document de type racine -->
      </racine>
      ```

- La syntaxe d'un fichier DTD est la suivante :
    - Pour un élément : <!ELEMENT name (model) >
        - ELEMENT est un mot-clé
        - model est le content model de l'élément et qui est spécifié en utilisant une expression régulière sur les noms d'éléments (exemple : (livre | article)* est un modèle qui signifie qu'on aura dans une bibliographie un livre ou bien un article. La signification de * est que zéro ou plusieurs occurrences peuvent avoir lieu)
- Les types prédéfinis utilisables sont les suivants :

Type prédéfini	Description
ANY	L'élément peut contenir tout type de données
EMPTY	L'élément ne contient pas de données spécifiques
#PCDATA	L'élément doit contenir une chaîne de caractères

- Ainsi un élément nommé Nom contenant un type #PCDATA sera déclaré de la façon suivante dans la DTD :

  <! ELEMENT Nom (#PCDATA) >

- D'autre part il est possible de définir des règles d'utilisation, c'est-à-dire les éléments XML qu'un élément peut ou doit contenir. Cette syntaxe se fait à l'aide de notations spécifiques dont voici un récapitulatif :

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 ou l'élément B peuvent être présents	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)+

- Ainsi on peut créer la déclaration suivante dans la DTD :

  <!ELEMENT personne (nom, prenom, telephone, email?) >

  - Pour un attribut, on définit le nom, le type, la valeur par défaut. Par exemple, <!ATTLIST rss version CDATA #FIXED "2.0" > : rss est l'élément auquel l'attribut est associé, version est le nom de l'attribut, CDATA est type de l'attribut, "2.0" est la valeur par défaut, #FIXED "x" si présent sinon sa valeur doit être x.

Il est possible d'ajouter des propriétés à un élément particulier en lui affectant un attribut, c'est-à-dire une paire clé/valeur. Ainsi avec XML la syntaxe pour définir un attribut est la suivante :

<! ATTLIST Elément Attribut Type >

Type représente le type de donnée de l'attribut, il en existe trois :

- littéral: il permet d'affecter une chaîne de caractères à un attribut. Pour déclarer un tel type il faut utiliser le mot clé CDATA
- l'énumération: cela permet de définir une liste de valeurs possibles pour un attribut donné, afin de limiter le choix de l'utilisateur. La syntaxe de ce type d'attribut est :

<! ATTLIST Elément Attribut (Valeur1 | Valeur2 | ...) >

Pour définir une valeur par défaut il suffit de faire suivre l'énumération par la valeur désirée entre guillemets :

<! ATTLIST Elément Attribut (Valeur1 | Valeur2) "valeur par défaut" >

- atomique: il permet de définir un identifiant unique pour chaque élément grâce au mot clé ID.

Enfin chacun de ces types d'attributs peut être suivi d'un mot clé particulier permettant de spécifier le niveau de nécessité de l'attribut :

- #IMPLIED signifie que l'attribut est optionnel, c'est-à-dire non obligatoire
- #REQUIRED signifie que l'attribut est obligatoire
- #FIXED signifie que l'attribut sera affecté d'une valeur par défaut s'il n'est pas défini. Il doit être immédiatement suivi de la valeur entre guillemets

Ainsi on pourra avoir une déclaration d'attribut du type :

<! ATTLIST disque
IDdisk ID #REQUIRED
Type (K7|MiniDisc|Vinyl|CD)"CD"
>

Ce qui signifie que l'on affecte à l'élément disque deux attributs IDdisk et type. Le premier attribut est de type atomique, il s'agit d'un identifiant unique obligatoire. L'élément type peut être soit K7, MiniDisc, Vinylou CD, sachant que ce dernier sera affecté par défaut...