INFO-H-509 : Technologies XML
TP 1 - Correction

Professeur : Stijn Vansummeren
Assistant : Michaël Waumans
http://cs.ulb.ac.be/public/teaching/infoh509

XML Well-formedness

```
<?xml version="1.0" encoding="utf-8" ?>
<catalogue>
    <course mnemonic="INFO-H-509">
        <title>Technologies XML</title>
        <professor email="stijn.vansummeren@ulb.ac.be">Stijn Vansummeren</professor>
        <assistant email="fpicalau@ulb.ac.be">François Picalausa</assistant>
        <weight>3 <unit type="ECTS" /></weight>
    </course>
    <course mnemonic="INFO-H-302">
        <!-- Ce cours ne sera plus donné l'année prochaine -->
        <title>Analyse &amp;amp; conception par object</title>
        <professor email="ezimanyi@ulb.ac.be">Esteban Zimányi</professor>
        <assistant email="fservais@ulb.ac.be">Frédéric Servais</assistant>
        <assistant email="sboucher@ulb.ac.be">Serge Boucher</assistant>
        <weight>3 <unit type="ECTS" /></weight>
    </course>
    <course mnemonic="INFO-H-303">
        <title>Bases de données</title>
        <professor email="ezimanyi@ulb.ac.be">Esteban Zimányi</professor>
        <assistant email="fservais@ulb.ac.be">Frédéric Servais</assistant>
        <weight>6 <unit type="ECTS"/></weight>
    </course>
    <person email="stijn.vansummeren@ulb.ac.be" laboratory="WIT">
        <name>Stijn Vansummeren</name>
        <interest>Systèmes d'information</interest>
        <interest>Bases de données scientifiques</interest>
        <interest>Web sémantique</interest>
    </person>
    <person email="ezimanyi@ulb.ac.be" laboratory="WIT">
        <name>Esteban Zimányi</name>
        <interest>Systèmes d'information</interest>
        <interest>Entrepôts de données</interest>
        <interest>Web sémantique</interest>
    </person>
    <person email="fpicalau@ulb.ac.be" laboratory="WIT">
        <name>François</name>
        <interest>Web sémantique</interest>
    </person>
```

```
</catalogue>
```

Il faut remarquer que les attributs ne peuvent pas être multivalués. De même un attribut ne peut pas être décomposé en plusieurs champs, comme c'est le cas pour une adresse. A l'exception de ces contraintes, le choix entre éléments et attributs est le plus souvent arbitraire.

Unicode

Exercice 1.4

Donner le caractère associé au *code point* U+265C (nom et glyphe).

Il s'agit d'un caractère représentant la tour noire aux échec (BLACK CHESS ROOK).

Exercice supplémentaire : Donner l'encodage UTF-8 de ce caractère.

Le code point nous indique que le caractère est encodé par 3 code units. De plus $265C_{16} = 10\ 0110\ 0101\ 1100_2$. On obtient donc l'encodage 11100010 10011001 10011100.

Exercice 1.5

Donner les code points encodés par la séquence UTF-8 `00110100 00110010 11100010 10000000 10100110`.

Lorsque le premier bit d'un code-unit est à 0, cela signifie qu'il encode seul un caractère. S'il commence par 1110, cela signifie qu'il fait partie d'un groupe de trois code units successifs qui encodent un seul caractère. Nous avons donc :

- Code unit 1 : $011\ 0100_2 = 34_{16}$ est le caractère "4",
- Code unit 2 : $011\ 0010_2 = 32_{16}$ est le caractère "2".
- Code units 3 à 5 : $0010\ 000000\ 100110_2 = 2026_{16}$ réfère au caractère HORIZONTAL ELLIPSIS "...".

Le texte complet est "42..."

Requêtes XPath

Exercice 1.6

1. `//Customer[@CustomerID="HUNGC"]//Country`
   Retourne le pays dans lequel est basé le client "HUNGC".
2. `/Root/Orders//ShipCountry | /Root/Customers//Country`
   L'ensemble des pays des clients et des pays de livraison.
3. `//ShipCity[following-sibling::ShipRegion eq "OR"]`
   Pour l'ensemble des livraisons en Oregon, retourne la ville de livraison.
4. `//ShipCity[following-sibling::ShipRegion is "OR"]`
   Erreur : l'opérateur is compare des noeuds. "OR" est un litéral (de type xs :string) et non un noeud.
5. `/Root/Customers/Customer/Phone[1]`
   Retourne le premier numéro de téléphone de chaque client.
6. `(/Root/Customers/Customer/Phone)[1]`
   Retourne le premier numéro de téléphone dans le document.

Exercice 1.7

Il est toujours plus efficace d'utiliser un chemin complet, plutôt que l'axe `descendant-or-self` ou son abréviation `//`, malgré la commodité de cette dernière. Pour des raisons de lisibilités, nous négligeons dans la suite cette règle de bonne pratique.

1. Les clients qui ont le titre de Marketing Manager.

   ```
   /Root/Customers/Customer[ContactTitle eq "Marketing Manager"]
   ```

   Remarque : On pourrait utiliser `=` au lieu de `eq`. Néanmoins, comme on sait que l'élement ContactTitle est unique pour un client, utiliser eq permet éventuellement au moteur XPath d'optimiser la requête.

2. Les éléments HTML du document.

   ```
   //*[namespace-uri() eq "http://www.w3.org/1999/xhtml"]
   ```

   Remarque : On peut aussi utiliser `//html:*`, à condition de définir dans le namespace correspondant dans le moteur XPath. En effet, le préfixe html déclaré dans le document est totalement indépendant des préfixes reconnus par le moteur XPath. Ceci permet que les expressions XPath restent valides même en cas de changement de préfixe dans le document.

3. Les clients dont le nom contient Yoshi.

   ```
   /Root/Customers/Customer[contains(CustomerName, "Yoshi")]
   ```

4. Les frais de transports les plus élevés, les moins élevés et moyens. (Element Freight)

   ```
   (max(//Freight), min(//Freight), avg(//Freight))
   ```

5. Les commandes qui n'ont pas encore été envoyées (attribut ShippedDate absent pour ShipInfo)

   ```
   //ShipInfo[not(@ShippedDate)]
   ```

6. La première commande (dans l'ordre du document), pour chaque client.

   ```
   //Order[not(preceding-sibling::Order/CustomerID = CustomerID)]
   ```

7. Les commandes réalisées par GREAL en Avril 1998.

   ```
   //Order[CustomerID="GREAL"]
          [xs:dateTime("1998-04-01T00:00:00") le xs:dateTime(OrderDate)]
          [xs:dateTime(OrderDate) lt xs:dateTime("1998-05-01T00:00:00")]
   ```

8. Les commandes réalisées par un client basé en Californie.

   ```
   //Order[CustomerID =
           /Root//Customer[FullAddress/Region eq "CA"]/@CustomerID]
   ```

9. La liste des pays de livraison ou, pour l'Amérique, des états.

   ```
   for $country in //ShipCountry return
       if ($country eq "USA") then
           $country/preceding-sibling::ShipRegion/text()
       else
           $country/text()
   ```

   Ou encore :

   ```
   //ShipInfo[ShipCountry eq "USA"]/ShipRegion/text() |
   //ShipInfo[ShipCountry ne "USA"]/ShipCountry/text()
   ```

10. La requête qui teste s'il existe une commande livrée par le livreur 3.

    ```
    some $x in //ShipVia satisfies $x=3
    ```