Table des matières

Documentation **libxml2** :
`http://xmlsoft.org/html/index.html`

Liste des traductions francaises des recommandations du W3C:
`http://www.w3.org/Consortium/Translation/French`

La librairie de développement libxml2

Vérifier la présence de la **libxml2** :

```
$ rpm -qa | grep libxml2
libxml2-2.6.13-1.1.101mdk
libxml2-devel-2.6.13-1.1.101mdk
libxml2-utils-2.6.13-1.1.101mdk
```

Sinon installer les paquetages **RPM**s.

Visualiser les fichiers de développement installés :

```
$ rpm -ql libxml2-devel-2.6.13-1.1.101mdk
/usr/bin/xml2-config              # script qui fournit des infos sur la libxml2
/usr/include/libxml2/libxml                  # les fichiers header
/usr/lib/libxml2.a                           # librairie statique
/usr/lib/libxml2.so                          #librairie dynamique
/usr/share/doc/libxml2-devel-2.6.13/         # la doc
... etc ...
```

Pour compiler un programme utilisant la **libxml2**, il faudra les options suivantes :

```
$ xml2-config --cflags
-I/usr/include/libxml2            # chemin d'inclusion des fichiers .h

$ xml2-config  --libs
-lxml2 -lz -lpthread -lm          # les librairies dynamiques pour l'édition de liens
```

Concrètement, on utilisera la ligne suivante pour compiler un programme testXML.c :

```
$ gcc testXML.c `xml2-config --cflags --libs` -o  testXML
```

TP 1 –Visualiser le document XML

1 . A partir d'un éditeur de texte, créer le document XML **livres.xml** :

```
<?xml version="1.0" encoding="UTF-8"?>
<livres>
<livre>
<titre>Webmaster in a Nutshell</titre>
<auteur>Stephen Spainhour</auteur>
<auteur>Robert Eckstein</auteur>
<codeisbn>2-84177-087-7</codeisbn>
<editeur>O'Reilly</editeur>
</livre>
<livre>
<titre>La bible du programmeur C/C++</titre>
<auteur>Kris Jamsa</auteur>
<auteur>Lars Klander</auteur>
<codeisbn>2-212-09058-7</codeisbn>
<editeur>Eyrolles</editeur>
</livre>
<livre>
<titre>Le langage C++</titre>
<auteur>Bjarne Stroustrup</auteur>
<codeisbn>2-7440-1089-8</codeisbn>
<editeur>Pearson Education</editeur>
</livre>
</livres>
```

2 . Ecrire et tester le programme **dumpXML.c** :

```
#include <stdio.h>
#include <libxml/parser.h>
#include <libxml/tree.h>

int main(int argc, char **argv)
{
    xmlDoc *doc = NULL;

    if (argc != 2)
        return(1);

    /*parse the file and get the DOM */
    doc = xmlReadFile(argv[1], NULL, 0);

    if (doc == NULL)
    {
        printf("Parse erreur ! (%s)\n", argv[1]);        return 1;
    }

    printf("Le fichier %s :\n", argv[1]);
    xmlDocDump(stdout, doc);
    printf("\n\n");

    /*free the document */
    xmlFreeDoc(doc);

    /*Free the global variables that may have been allocated by the parser. */
    xmlCleanupParser();

    return 0;
}
```

TP 2 –Lire les données du document XML

1 . Ecrire et tester le programme **affichXML.c** :

```
#include <stdio.h>
#include <libxml/parser.h>
#include <libxml/tree.h>

// cette fonction (récursive) affiche le nom de tous les éléments du document XML
static void print_element_names(xmlNode *a_node)
{
    xmlNode *cur_node = NULL;

    for(cur_node = a_node; cur_node; cur_node = cur_node->next)
    {
        if (cur_node->type == XML_ELEMENT_NODE)
       {
            printf("node type: Element, name: %s\n", cur_node->name);
        }
        print_element_names(cur_node->children);
    }
}

int main(int argc, char **argv)
{
    xmlDoc *doc = NULL;
    xmlNode *root_element = NULL;

    if (argc != 2)
        return(1);

    doc = xmlReadFile(argv[1], NULL, 0);

    if (doc == NULL)     { printf("Parse erreur ! (%s)\n", argv[1]);           return 1;
}

    /* Get the root element node */
    root_element = xmlDocGetRootElement(doc);

    print_element_names(root_element);

    xmlFreeDoc(doc);
    xmlCleanupParser();
    return 0;
}
```

2 . Modifier le programme précédent (en fait le fonction `print_element_names()`) afin d'afficher les données du document XML :

- le type à détecter est **`XML_TEXT_NODE`**
- les données sont dans **`cur_node->content`**

Remarque: éviter d'afficher les sauts de lignes contenus dans le fichier **livres.xml**

TP 3 –Ajouter des données dans un document XML

1 . Compléter et tester le programme **ajoutXML.c** :

```
#include <stdio.h>
#include <libxml/parser.h>
#include <libxml/tree.h>

int main(int argc, char **argv)
{
    xmlDoc *doc = NULL;
    xmlNode *root_node = NULL, *node = NULL;

    if (argc != 2)
        return(1);

    doc = xmlReadFile(argv[1], NULL, 0);

    if (doc == NULL)
    {
        printf("Parse erreur ! (%s)\n", argv[1]); return 1;
    }

    root_node = xmlDocGetRootElement(doc);
    /* Associate the root_node with document */
    xmlDocSetRootElement(doc, root_node);

    /* Add new sub node */
    node = xmlNewNode(NULL, BAD_CAST "livre");
    xmlNewChild(node, NULL, BAD_CAST "titre", BAD_CAST "Java");
    // ... etc ... : à compléter

    xmlAddChild(root_node, node);

    /* Saving the file */
    xmlSaveFormatFileEnc(argv[1], doc, "UTF-8", 1);

    xmlFreeDoc(doc);
    xmlCleanupParser();
    return 0;
}
```

