

# Bachelier en Informatique de Gestion

## Web : principes de base Projet de Développement Web

Enseignement supérieur économique de type court

Code FWB : 7534 29 U32 D1, 7534 30 U32 D3

Code ISFCE : 4IWPB, 4IPW3



# Table des matières

## Généralités

01. Introduction au web

03. Outils

05. Format XML

06. Format JSON

## Front-End

12. Structure HTML

13. Formulaire HTML

14. Mise en forme CSS

15. Adaptabilité

17. Javascript

18. Bibliothèque jQuery

19. Composant Vue.js

## Back-End

21. Middleware PHP

22. Traitement du  
formulaire

23. Architecture MVC

24. Données SQL

25. Données NoSQL

27. Requête asynchrone



## 05. Format XML

SGML

HTML

XML

XHTML

DTD

XSD

XSL

Xpath

Xquery



= BUROTIX ()

# XML : description

SGML

DTD

HTML

XSL

XML

XSD



# XML en quelques mots

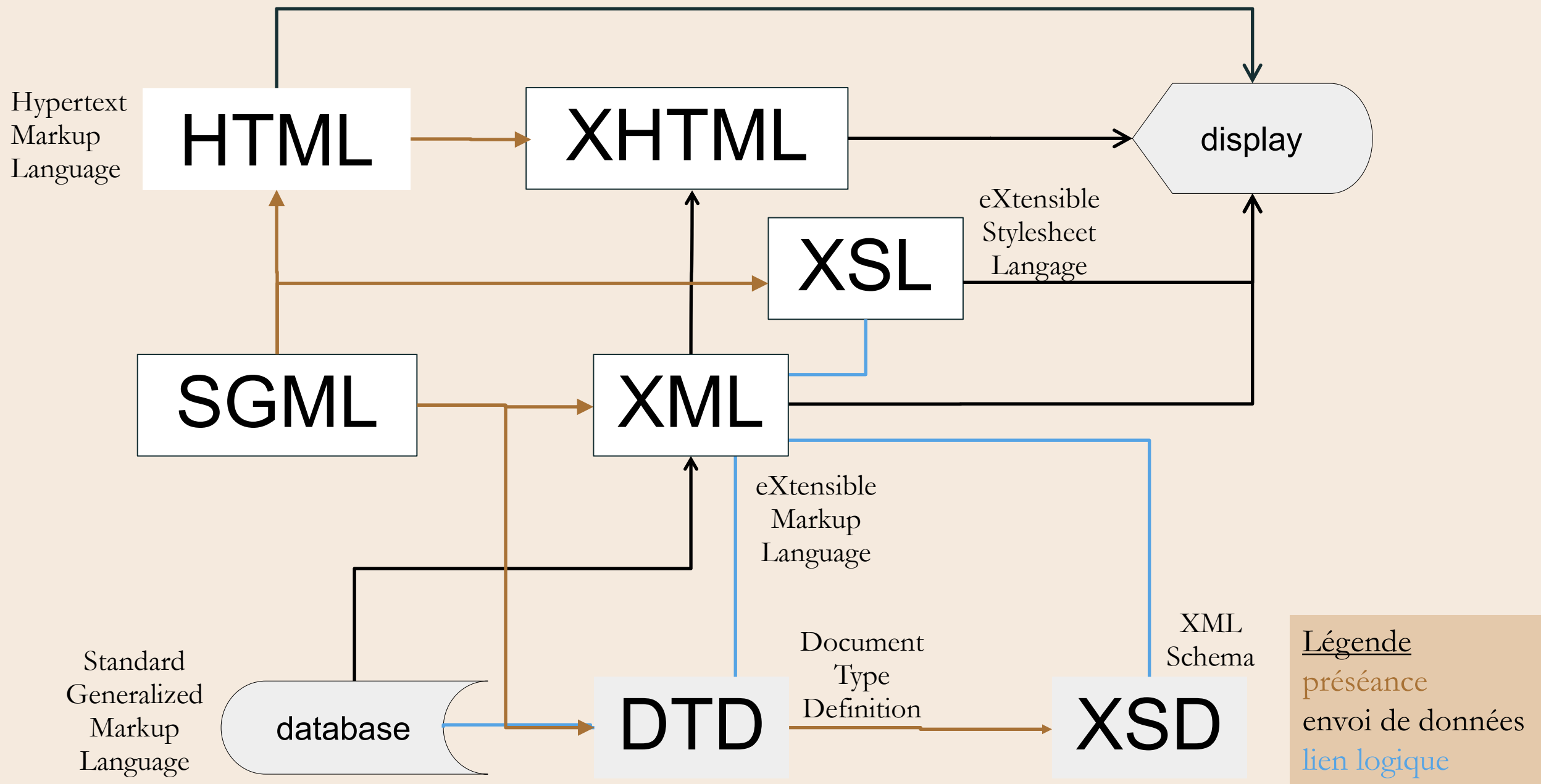
- Extensible Markup Language
  - *"langage de balisage extensible"* (chevrons)
  - Méta-langage informatique de balisage générique
- Syntaxe « extensible » car elle permet
  - Différents vocabulaires
  - Différentes grammaires
- Objectif : interopérabilité
  - Échange automatisé de contenus entre systèmes d'informations
  - Lisibilité humaine

```
<?xml version="1.0"?>
<questionnaire>
  <question>
    Qui était le premier
    empereur romain ?
  </question>
  <réponse>
    Auguste
  </réponse>
  <!-- Note : tu auras besoin
    de plus de questions.-->
</questionnaire>
```

**XML**



= BUROTIX ()



# La famille XML

- SGML

- Langage de description à balises (1986)
- Orientation « document »
- Riche, complet, lourd : 155 pages specs

- HTML

- Orientation « document web »
- Simple mais limité au web

- XML

- Orientation « données »
- SGML simplifié : 35 pages specs
- Souplesse SGML + Simplicité HTML
- Présentation des données : out of scope

- XHTML

- HTML suivant la norme XML

- Les adjoints du XML :

- DTD : la structure du document, permettant sa validation
- XSD : le DTD en plus évolué
- XSL : pour transformer XML
- Namespace, Xpath, Xquery, etc.

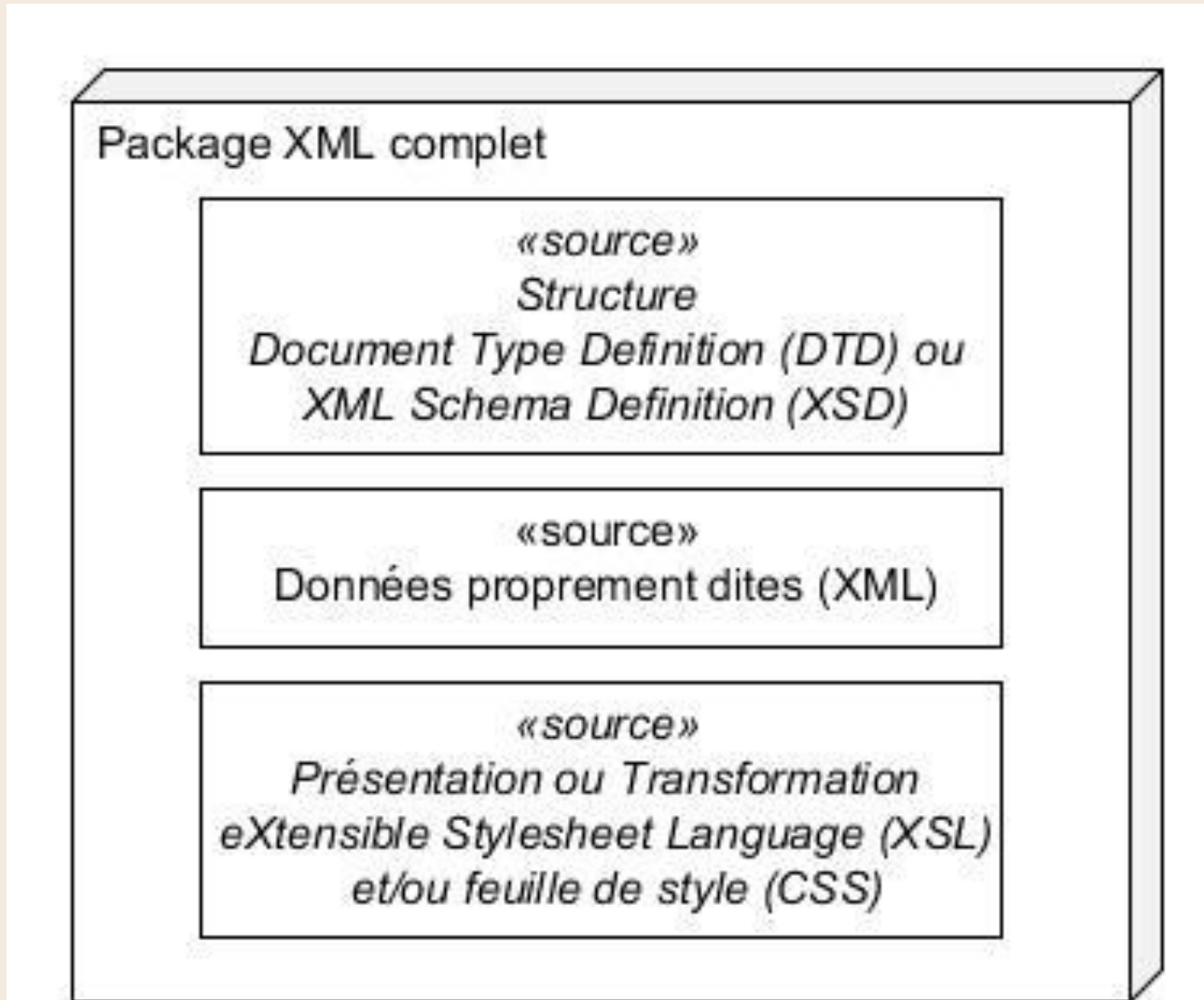


# Ensemble complet de données XML

- XML
  - données proprement dites
  - balise ouvrante – contenu – balise fermante
  - balises encastrables
- DTD ou XSD
  - Grammaire du doc XML
  - Validation des données échangées
  - DTD/XSD + XML = données validées
  - Élément optionnel.
  - Application : Des partenaires s'échangeant des données par XML
- DTD obsolète
- XSL et CSS
  - Pour présenter un doc XML
  - un XSLT par type de présentation
    - Un pour Web, un autre pour Mobile, un pour PDF, un pour l'écran textuel monochrome.
  - Élément optionnel.
  - XML + XSL (+ CSS) = doc lisible



# Ensemble complet de données XML



# XML, exo 01

```
<?xml version="1.0"  
      encoding="ISO-8859-1"  
      standalone="no" ?>
```

```
<?xml-stylesheet  
      type="text/xsl"  
      href="bonjour.xsl" ?>
```

```
<!DOCTYPE salut SYSTEM "bonjour.dtd">
```

```
<salut>Hello world!</salut>
```

Entête

Réf. XSL

Réf. DTD

Arbre des données

Prologue

# DTD, exo 01

Une balise « salut » peut contenir n'importe quelle valeur.

`<!ELEMENT salut (#PCDATA) >`

La structure XML peut contenir des balises de nom « salut ».



# DTD, exo 01

- Un élément est défini par
  - l'expression **<!ELEMENT**
  - le nom de l'élément, par exemple **salut**
  - des mentions sur le contenu du type d'élément, qui peuvent être très compliquées.
  - l'expression **>**
- Un type de contenu très populaire :
  - **(#PCDATA)** : *parsed character data*, càd ... du texte (chaîne de caractères alphanumériques), càd « autant de texte que vous voulez mais aucun élément intérieur »

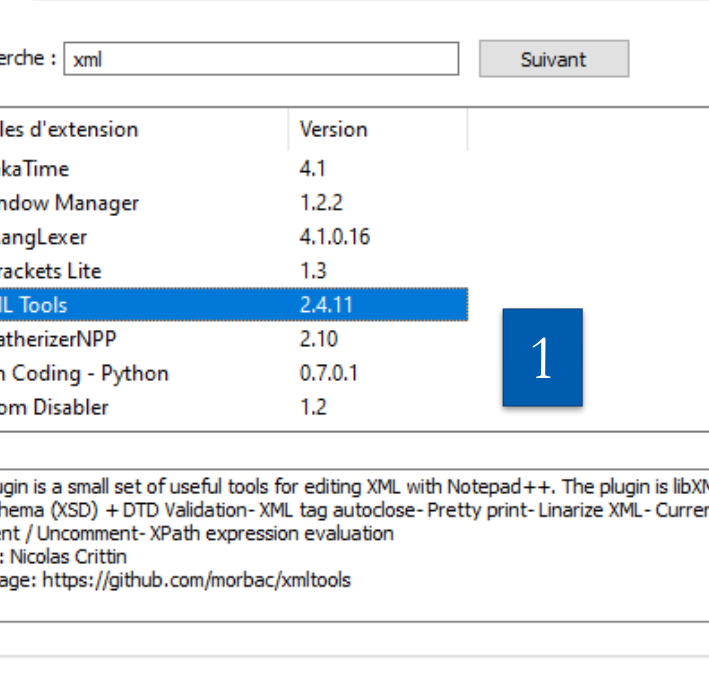


# Outils

- Notepad++
  - Extension "XML Tools" à installer
- MS XML Notepad
  - <http://www.lovetsoftware.com/downloads/xmlnotepad/readme.htm>
- OXYGEN
  - <https://www.oxygenxml.com>
- ONLINE XML EDITOR
  - [https://www.tutorialspoint.com/online\\_xml\\_editor.htm](https://www.tutorialspoint.com/online_xml_editor.htm)



# Outils : Notepad++



Gestionnaire des modules d'extension

Disponibles Mises à jour Installés

Recherche : xml Suivant

Modules d'extension	Version
<input type="checkbox"/> WakaTime	4.1
<input type="checkbox"/> Window Manager	1.2.2
<input type="checkbox"/> WLangLexer	4.1.0.16
<input type="checkbox"/> XBrackets Lite	1.3
<input checked="" type="checkbox"/> XML Tools	2.4.11
<input type="checkbox"/> XPatherizerNPP	2.10
<input type="checkbox"/> Zen Coding - Python	0.7.0.1
<input type="checkbox"/> Zoom Disabler	1.2

1

This plugin is a small set of useful tools for editing XML with Notepad++. The plugin is libXML2-based. XML Schema (XSD) + DTD Validation - XML tag autoclose - Pretty print - Linarize XML - Current XML Path Comment / Uncomment - XPath expression evaluation  
 Author: Nicolas Crittin  
 Homepage: <https://github.com/morbac/xmltools>

Fermer

The screenshot shows the Notepad++ application with the XML Tools menu open. The menu options are as follows:

- Enable XML syntax auto-check
- Check XML syntax now
- Enable auto-validation
- Validate now (Ctrl+Alt+Shift+M)
- Tag auto-close (highlighted with a blue box and the number 2)
- Set XML type automatically
- Prevent XXE
- Allow huge files
- Pretty print (XML only)
- Pretty print (XML only - with line breaks) (Ctrl+Alt+Shift+B)
- Pretty print (Text indent)
- Pretty print (libXML) [experimental]
- Pretty print (attributes) (Ctrl+Alt+Shift+A)
- Linearize XML (Ctrl+Alt+Shift+L)
- Apply to all open files
- Current XML Path (Ctrl+Alt+Shift+P)
- Current XML Path with predicates
- Evaluate XPath expression...
- XSL Transformation...
- Convert selection XML to text (<> => &lt;&gt;)
- Convert selection text to XML (&lt;&gt; => <>)
- Comment selection (Ctrl+Alt+Shift+C)
- Uncomment selection (Ctrl+Alt+Shift+R)
- Options...
- About XML Tools / Donate...

The background XML file content is partially visible, showing a DTD and a list of books within a <biblio> tag.

# XML, exo 02

library+dtd.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<biblio>
  <livre>
    <!-- premier élément de type livre -->
    <titre>Les Misérables</titre>
    <auteur>Victor Hugo</auteur>
    <nb_tomes value="3" />
  </livre>
  <livre>
    <titre>L'Assomoir</titre>
    <auteur>Émile Zola</auteur>
    <couverture couleur="rouge" />
  </livre>
  <livre lang="en" type="roman">
    <titre>David Copperfield</titre>
    <auteur>Charles Dickens</auteur>
    <nb_tomes value="2" />
  </livre>
</biblio>
```

# DTD, exo 02

library.dtd

```
<!ELEMENT biblio (livre*)>

<!ELEMENT livre (titre, auteur, nb_tomes?, couverture?)>
  <!ATTLIST livre
    type (roman | nouvelles | poemes | théâtre) #IMPLIED
    lang CDATA "fr"
  >

<!ELEMENT titre (#PCDATA)>

<!ELEMENT couverture EMPTY>
  <!ATTLIST couverture
    couleur (rouge | vert | jaune | bleu ) #IMPLIED
  >

...
```



= BUROTIX ()



# Schémas (XSD)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="contacts" type="typeContacts" />
  <xsd:element name="remarque" type="xsd:string">
    <xsd:complexType>
      <xsd:attribute
        name="maj"          type="xsd:date"
        use="optional"      default="2014-03-01"
      />
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Pour ce XML :

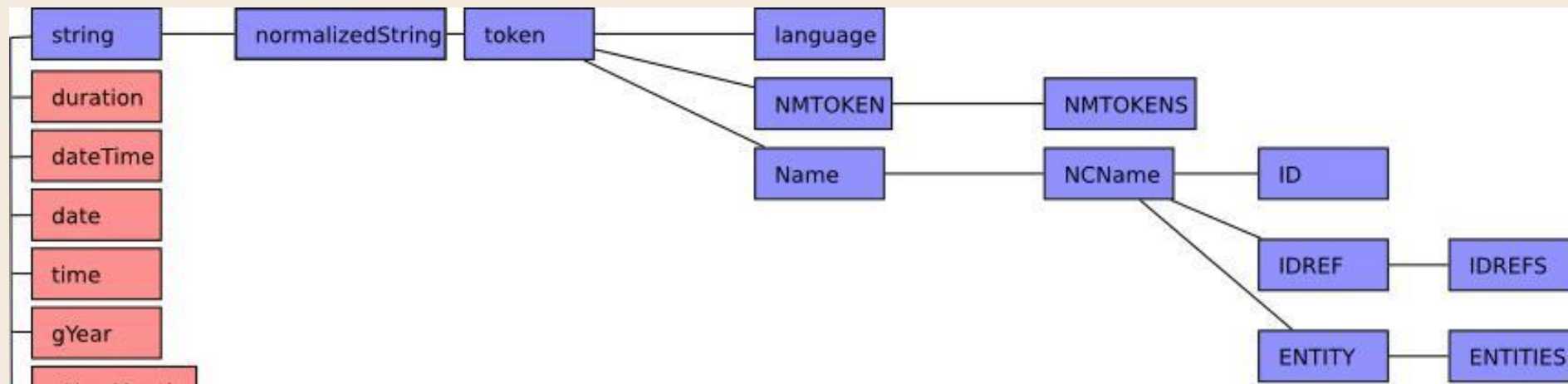
```
<contacts> ... </contacts>
<remarque maj="2014-03-08">bla bla</remarque>
```



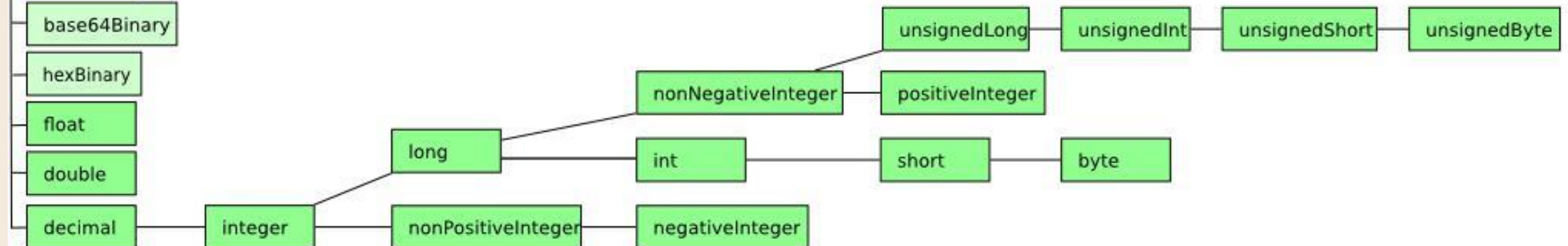
= BUROTIX ()

# Schémas (XSD) vs. DTD

- Syntaxe XML
- Typage des données
  - booléens, entiers, intervalles de temps, etc.,
  - **dérivation** : types nouveaux à partir de types existants
- Propriétés plus précises
  - Par exemple : min, max
- **Élément** : toujours "typé"
  - Soit de type simple : xsd:string , xsd:date, xsd:boolean, etc.
  - Soit de type **complexe** (défini par l'utilisateur) : par ex. typeContacts
- **Attribut**
  - toujours de type simple.
- **Dérivation**
  - Restriction appliquée à un type simple ou complexe
  - Par exemple : type «MonEntier» = type « Integer », limité aux valeurs comprises entre 0 et 99.
- **Héritage** : les éléments peuvent hériter du contenu et des attributs d'un autre élément.
- **Espaces de nom** : pour utiliser plusieurs schémas indépendants dans un document XML
- **Modularité**
  - Plusieurs schémas possibles pour un même doc XML



## Schémas (XSD) : types simples prédéfinis



# eXtensible Stylesheet Language (XSL)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <head>
        <title>Contenu de ma bibliothèque</title>
        <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
        <link rel="stylesheet" type="text/css" href="library.css" />
      </head>
      <body>
        <h1>Contenu de ma bibliothèque</h1>
        <!-- voir slide suivant -->
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

# eXtensible Stylesheet Language (XSL)

- Manipulation de **templates**
  - format de sortie
  - précisant où les éléments XML sont placés
- Sous format XML
- XSL = trois langages :
  - **Xpath**
    - Pour accéder à un nœud quelconque de l'arborescence d'un document XML
  - **XSLT** - eXtensible Stylesheet Language Transformation
    - pour transformer un doc XML source en un doc XML cible
    - en bouleversant sa structure éventuellement
- **XSL-FO** - eXtensible Stylesheet Language - Formatting Objects
  - tout langage permettant la mise en page finale
  - application principale : PDF.
- Structure de base :
  - un **prologue** « xml »
  - un élément **xsl:stylesheet**
    - attributs : notamment une déclaration d'espace de noms
    - élément racine du document XSL.
  - un élément **xsl:template**, enfant de **xsl:stylesheet**, contenant la structure de l'output



# eXtensible Stylesheet Language (XSL)

```
<xsl:for-each select="biblio/livre">
  <div class="livre">
    <div class="titre">
      "<xsl:value-of select="."/titre" />"
    </div>
    <div class="auteur">
      Auteur : <xsl:value-of select="./auteur" />
      <xsl:if test="./nb_tomes">
        <div>
          Nombre de tomes :
          <xsl:value-of select="./nb_tomes" />
        </div>
      </xsl:if>
      <xsl:if test="@lang='en'">
        <div class="attention">
          Attention : Ce livre est en Anglais.
        </div>
      </xsl:if>
    </div>
  </div>
</xsl:for-each>
```

# eXtensible Stylesheet Language (XSL)

- Quelques éléments intéressants des XSL :
  - Xpath : `"/titre", "biblio/livre", "@lang='en'"`  
pour accéder à un endroit quelconque de l'arbre XML  
similaire à une query sql
  - `<xsl:value-of select="/titre" />`  
retourne la (les) valeurs sélectionnées par le Xpath
  - `<xsl:for-each select="biblio/livre"> ... </xsl:for-each>`  
lance une boucle sur les éléments retournés par le Xpath
  - `<xsl:if test="@lang='en'"> ... </xsl:if>`  
lance un test booléen sur l'expression Xpath indiquée en test



# eXtensible Stylesheet Language (XSL)

- Remarques sur l'exemple donné

- Mixage de balises
  - XSL : **for-each**
  - HTML : **div**
- Possible de mixer des balises de différents *namespaces*.
- Namespace « **xsl** » qui

préfixe les balises XSL

- **xsl:for-each**
- **xsl:if** ...
- Namespace HTML ?
  - Les balises HTML ne sont pas préfixées !
  - Pas de préfixe comme **html:h1**, **html:div**, ...





# Exo 03

- Fichiers
  - `library+dtd+xsl.xml`
  - `exo03_[abc].xsl`
  - `library.dtd`
- Installez-les et visualisez-les grâce à WAMP
- Différenciez les trois XSL
  - `a = ?` `b = ?` `c = ?`
- Adaptez l'ensemble pour pouvoir introduire le livre suivant:
  - Titre : "Das Parfum"
  - Sous-titre : "Die Geschichte eines Mörders"
  - Auteur : Patrick Süskind
  - Langue : allemande

## Contenu de ma bibliothèque

Titre	Auteur	Nombre de tomes	Langue
"Les Misérables"	Victor Hugo		
"L'Assomoir"	Emile Zola		
	Dickens		en
	van den Vondel		nl

## Contenu de ma bibliothèque

"Les Misérables"      "L'Assomoir"  
Auteur : Victor Hugo    Auteur : Emil  
Pays : FR                Pays : FR

"Gisbrecht van Aemst"  
Auteur : Joost van der  
Attention : livre en Ne  
Pays : NL

## Contenu de ma bibliothèque

"Les Misérables"  
Auteur : Victor Hugo  
Pays : FR

"L'Assomoir"  
Auteur : Emile Zola  
Pays : FR

"David Copperfield"  
Auteur : Charles Dickens  
Attention : livre en Anglais.  
Pays : GB

"Gisbrecht van Aemstel"  
Auteur : Joost van den Vondel  
Attention : livre en  
Neerlandais.  
Pays : NL

```
<?xml version="1.0"?>
<biblio>
  <livre identifiant="_1234" nb_tomes="3">
    <!-- premier élément de type livre -->
    <titre>Les Misérables</titre>
    <auteur country="FR">Victor Hugo</auteur>
  </livre>
  <livre identifiant="_1235">
    <!-- 2e élément de type livre -->
    <titre>L'Assomoir</titre>
    <auteur country="FR">Emile Zola</auteur>
  </livre>
  <livre identifiant="1236" type="roman" lan
```

a ?

b ?

c ?



= BUROTIX ()

# XML : références



= BUROTIX ()

# Références

- w3school - une bonne adresse ;-)
  - <https://www.w3schools.com/xml/>
- phpnet.org
  - <https://www.php.net/manual/fr/function.simplexml-load-file.php>

