

**Boucle récursive de parcours de branche:**

```
<xsl:template name="boucleRecursive">
  <xsl:param name="Node"/>
  ...
  <xsl:if test="$Node/following-sibling::*">
    <xsl:call-template name="boucleRecursive">
      <xsl:with-param name="Node"
        select="$Node/following-sibling::*"/>
    </xsl:call-template>
  </xsl:if>
</xsl:template>
```

Xsl number (value-of return like):

```
<xsl:number level="single" ou multiple"
  count="nodeName [|UnionNodeName]"
  from="StartNodeXPath" format="A.1" />
```

Fonctions XSL 2.0+ déclaration:

```
<xsl:function name="Namespace:functionName">
  <xsl:param name="param1"/>
  <xsl:param name="param2">
    [select="XPathExpression"]/
  <!-- Contenu de la fonction-->
  <!-- retour de fonction par xsl:value-of -->
</xsl:function>
```

Fonctions 2.0 usage :

```
<xsl:value-of select="Namespace:functionName
  (param1,param2)"/>
```

HTML**Structure minimum Html :**

```
<html>
  <head>
    <title>Titre de votre page</title>
  </head>
  <body> <!-- Contenu de page -->
</html>
```

**Intégration fichier css (dans <head>) :**

```
<link href="fichier.css" rel="stylesheet" type="text/css">
```

Références balises Html :

Assignation class	<balise class="class1 [cl2 ...]"/>
Assignation id	<balise id="idName"/>
Elément bloc	<div>...</div>
Elément enligne	...
Tableau, ligne, cellule	<table><tr> <td>cellule</td> </tr></table>
Lien	visuel
image	
Liste de puces et puces	Une puce
Formulaire, entrée, soumission	<form method="POST GET" action="lienTraitement"> <input type="text password checkbox hidden radiobutton" name="Nom"> <input type="submit" value="Valider"> </form>
Paragraphe	<p> contenu paragraphe </p>
Titres niveau 1 (fort) à 6 (faible)	<h1>...</h1>, ... <h6>...</h6>

CSS**Syntaxe fichier CSS ou enligne html:**

```
sélecteur { propriété : valeur ; / propriété : valeur ; / }
<balise style="propriété : valeur ; / propriété : valeur ; /">
```

Sélecteurs simples CSS :

•NomClass → Classe réutilisable dans la page
 #NomIdentifiant → Identifiant à usage unique par page
 NomDeBalise → Balise portant NomDeBalise

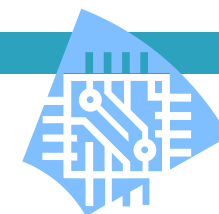
Sélecteur encapsulé :

SelecteurConteneur SelecteurContenu

Sélecteur multiple :

Selecteur1, Selecteur2, ...

Liste de propriétés : margin[-left|-right|-top|-bottom], padding[-left|-right|-top|-bottom], left, right, top, bottom, text-align, color, background[-color|-image|-attachment|-position], opacity, [min-|max-]width, [min-|max-]height, display, visibility, cursor, z-index, float, overflow, border[-width|-color|-style], ...



MEMENTO

piwya

XML, XPath, Xsl, CSS

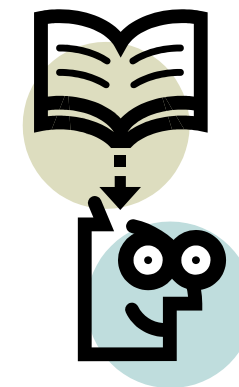


Les mémentos Programs Is What You Are, ou piwya, sont une suite d'outils d'aide au développement informatique.

Le memento XML, XPath, XSL, CSS est le rassemblement de tout ce qu'il est nécessaire de savoir sur ces

technologies, dans un format de poche transportable partout, et fait pour faciliter la vie du développeur XML et dérivés.

Avant toute chose, ce guide est un outil pédagogique pour progresser en Xsl dès les premiers pas, mais il convient tout a fait aux développeurs plus aguerris.



Section : Web XML développement

A.Desorbaix



XML, XPath	Xpath, fonctions préfixe : fn:*	Xsl	Xsl
<p>Règles XML :</p> <ul style="list-style-type: none"> - 1 seule balise Root - Pas de balise ouverte - Pas de balises croisées - nom d'attribut, nom de balises, valeurs d'id commencent par une lettre <p>XPath</p> <p>Sélecteurs :</p> <p>nodeName → sélectionne tous les nœuds fils du nœud</p> <p>/ → nœud racine</p> <p>// → tous les nœuds depuis le contexte.</p> <p>. → nœud courant</p> <p>.. → nœud parent du nœud courant.</p> <p>@ → attribut</p> <p>* → n'importe quel nœud élément</p> <p>@* → n'importe quel nœud attribut</p> <p>text() → n'importe quel nœud texte</p> <p>node() → n'importe quel nœud</p> <p>expr[exprboo] → expr : [n] à nième élément [exprBool] vérifiée</p> <p> (pipe) → UNION</p> <p>last() → renvoient le dernier</p> <p>position() → renvoient la position courante</p> <p>Axes :</p> <p>self → nœud courant</p> <p>ancestor[-or-self] → Ancêtre [ou courant]</p> <p>descendant[-or-self] → Descendant [ou courant]</p> <p>child, parent → enfant, parents</p> <p>following[-sibling] → suivant [même fratrie]</p> <p>preceding[-sibling] → suivant [même fratrie]</p> <p>namespace → espace de nom</p> <p>Axes usage :</p> <p>cheminXPath/axeName::nodeName/fils</p>	<p>Evaluation XPath dans un attribut de sortie :</p> <p><BaliseHtml attribut="{ExpressionXPath}" /></p> <p>Fonctions Boolean:</p> <p>Boolean(arg) → cast, true() à vrai, false() → faux</p> <p>not(boolean) → renvoie l'inverse de boolean</p> <p>Fonctions Math:</p> <p>Number(arg) → cast abs(num) → valeur absolue</p> <p>floor(num) → Entier inf. ceiling(num) → Entier sup.</p> <p>round(num) → Arrondi entier plus proche</p> <p>Fonctions String:</p> <p>concat(arg1,arg2) → concaténation de arg1 et arg2</p> <p>compare(str1,str2) → Compare 2 chaînes str1 et str2</p> <p>substring(str ,s[,L]) → Coupe str de s [sur L caractères]</p> <p>string-length(str) → renvoie la taille de str</p> <p>contains(str,str2),match(str,str2) → Vrai si str2 dans str</p> <p>end-with(str,str2)*, start-with(str,str2) → Vrai si str débute ou finie par str2</p> <p>Fonctions agrégat 1.0 & 2.0:</p> <p>min(Node) → min* max(Node) → max*</p> <p>count(Node) → Nombre d'éléments Node</p> <p>sum(Node) → Somme des éléments Node</p> <p>avg(Node) → Moyenne des éléments Node*</p> <p>Fonctions de nœuds :</p> <p>position() → Renvoie la position du contexte</p> <p>last() → position du dernière éléments du contexte</p> <p>Opérateurs :</p> <p>arithmétiques → + - div * mod</p> <p>logique → = != < > <= >= or and </p> <p><i>*uniquement xsl 2.0+</i></p>	<p>Liaison XML/XSL(text/Xsl) ou Css(text/css) :</p> <p><?xml-stylesheet type="text/ext" href="fichier.ext"?></p> <p>Commentaires :</p> <p><xsl:comment> Commentaires </xsl:comment></p> <p>Balise racine :</p> <p><xsl:stylesheet version="1.0 2.0 3.0 4.0" xmlns:xsl="http://www. w3.org/1999/XSL/Transform" ></p> <p>Balise type de sortie :</p> <p><xsl:output method="xml html text" [indent="yes no"] [omit-xml-declaration="yes no"] /></p> <p>Balise d'affichage de valeur :</p> <p><xsl:value-of select="XPath expression ou Node"/></p> <p>Expression XPath dans un attribut de balise finale:</p> <p><balise attr="{XPathExpr ou XPathNode}" /></p> <p>Balise de test simple :</p> <p><xsl:if test="Bool XPath exp">...</xsl:if></p> <p>Test complexe :</p> <p><xsl:choose></p> <p><xsl:when test="Bool XPath exp">...</xsl:when></p> <p><xsl:otherwise>...</xsl:otherwise></p> <p></xsl:choose></p> <p>Boucles:</p> <p><xsl:for-each select="XPathExpression"></p> <p><.../></p> <p></xsl:for-each></p> <p>Balise génération élément & attribut :</p> <p><xsl:element name="ElementName"></p> <p><xsl:attribute name="AttributName"></p> <p>ValAttribut</p> <p></xsl:attribute></p> <p>ValElement</p> <p></xsl:element></p>	<p>Include prioritaire de bibliothèques Xsl :</p> <p><xsl:import href="fichier.xml"/></p> <p>Import de bibliothèques Xsl :</p> <p><xsl:import href="fichier.xml"/></p> <p>Templates match :</p> <p><xsl:template match="CheminNode.XPath" > ... </xsl:template></p> <p>Variables const et usage :</p> <p><xsl:variable name="varName" select="XPathExpression"/></p> <p><xsl:value-of select="\$varName"/></p> <p>Template match & apply:</p> <p><xsl:template match="XPathNodeName"></p> <p>...</p> <p></xsl:template></p> <p><xsl:apply-template match="XPathNode"/></p> <p>Template nommé , param [valeur par défaut] :</p> <p><xsl:template name="tplName"></p> <p><xsl:param name="prmName" [select="XPathExpression"] /></p> <p><xsl:value-of select="\$prmName"/> etc. ...</p> <p></xsl:template></p> <p>Template nommé sans [avec] param appels :</p> <p><xsl:call-template name="tplName"></p> <p>[<xsl:with-param name="prmName" [select="XPathExpression"/>>]]</p> <p></xsl:call-template></p> <p>Multi-XML :</p> <p><xsl:variable name="xmlDoc" select="Document('fichier.xml')"/></p> <p><xsl:value-of select="\$xmlDoc/nodeName"/></p> <p>Trie par nœud :</p> <p><xsl:sort select="NodeNameTrie"/></p>
1	2	3	4