XPath 2.0

- XPath 2.0 travaille sur un modèle d'un document XML
- Tout résultat d'une expression XPath 2.0 est une **séquence**
- Possibilité d'usage de variables, d'expressions conditionnelles ou itératives et de fonctions.
  - Mais ce n'est pas un langage de programmation, il est utilisé au sein d'autres langage (XSLT, Xquery, …)

XPath 2.0

Expressions itératives

- Limite : ne fonctionne qu'à un seule niveau d'imbrication (utiliser XQuery pour aller plus loin)
- Exp:
  - for $n in //Personne[Prenom] return concat($n/Prenom, ' ', $n/Nom)
  - for $i in //Personne/Prenom return substring($i, 1, 1)
  - for $n in //Personne[Prenom], $i in $n/Initiale return concat($n/Prenom, ' ', $i, $n/Nom)

XPath 2.0

La **séquence**

- Tout résultat d'une expression XPath 2.0 est une séquence
  - d'éléments **typés** selon le modèle de données XDM
  - de pointeurs de nœuds
  - de possibles duplicata : ($node-set, $node-set)
- Par opposition à XPath 1.0
  - Les résultats étaient des *ensembles(donc sans ordre et sans duplicata) de nœuds.*

XPath 2.0

Test sur les types :

- //element(*, PersonneT)
  - signifie trouver tous les nœuds du type PersonneT (ou d'un sous-type)
- //element(*, xs:string)
  - attention le namespace doit être déclaré dans le document

XPath 2.0

Il reste toujours les tests sur les noms

- //element(Personne)
  - signifie trouver les nœuds de nom Personne

…mais aussi sur les éléments du schéma

- //schema-element(Annuaire)
  - utile pour les substitution groups

On peut combiner les deux (test sur nom et type)

- //element(Personne, PersonneT)

XPath 2.0

- Les expressions comme chemin
  - //(Nom | Prenom)/text()

  au lieu de
  - //Nom/text() | //Prenom/text()

  Sert pour régler le problème d'explosion combinatoire d'expressions logiques imbriquées

- Une gestion facilitée des namespaces
  - //*:Personne fonctionne

XPath 2.0

Les séquences

- Quelques exemples
  - la racine d'un document (donc un document)
  - un nœud (donc un sous arbre)
  - une série de nœuds et/ou de documents
  - une chaîne de caractères ou un entier
  - une série d'entiers
  - un ensemble de nœuds
  - tout résultat d'une expression XPath 2.0

XPath 2.0

La construction d'une séquence avec l'opérateur **,**

- (1, 2, 3) est une séquence de 3 entiers (xs:integer)
- (1, ‘‘a’’, 3) est une séquence d'un entier, d'un xs:string et d'un entier
- si on ne veut pas avoir le type par défaut, on peut utiliser un *cast : (1, xs:double(2))*
- Une séquence n'a qu'un seul niveau
  - (a, b, (c, d), b) == (a, b, c, d, b)
- Tout ce qui n'est pas une valeur atomique (simpleType) est séquence : un singleton = une séquence de 1 *item*

XPath 2.0

Construction d'une séquence

- En utilisant des itérations
  - (1 to 5) = (1, 2, 3, 4, 5)
  - reverse (1 to 5) = (5, 4, 3, 2, 1)
- En utilisant Xpath
  - (Personne, Annuaire)[descendant::Prenom]
- En utilisant une expression ***for***
- En utilisant ***some*** ou ***every***

XPath 2.0

Les types

- XPath 1.0 avait 4 types : string, number, boolean et nodeSet
- XPath 2.0 peut utiliser tous les types (simple et complexe) de XMLSchema + tous les types définis dans n'importe quelle instance de XMLSchema

➔Les fonctions font désormais du typechecking de leurs paramètres.

XPath 2.0

Les items

- Un *item* est soit une valeur, soit est une référence d'un nœud
- Un *item* possède une valeur et un type

Une séquence est un ensemble ordonné d'items