haute école neuchâtel berne jura arc gestion neuchâtel delémont

Hes·so Haute Ecole Spécialisée de Suisse occidentale

Certifié ISO 9001

62-31.3 XML

XML-02-P02-Types_Predefinis_XSD

Cédric Benoit

Plan du cours

- Présentation et organisation
- Introduction à XML
- Schémas XML
- **Types de données**
  - **Types prédéfinis**

Importance du typage des données

Jusqu'à maintenant, nous avons étudié:

- Comment élaborer un document XML
- Comment définir la structure du document XML (XSD)

Pour aller plus loin dans la validation du document XML, nous allons attribuer à chaque donnée un type de donnée adéquat.

Typage de la donnée → Améliorer la qualité et la précision de la donnée, afin de la traiter correctement.

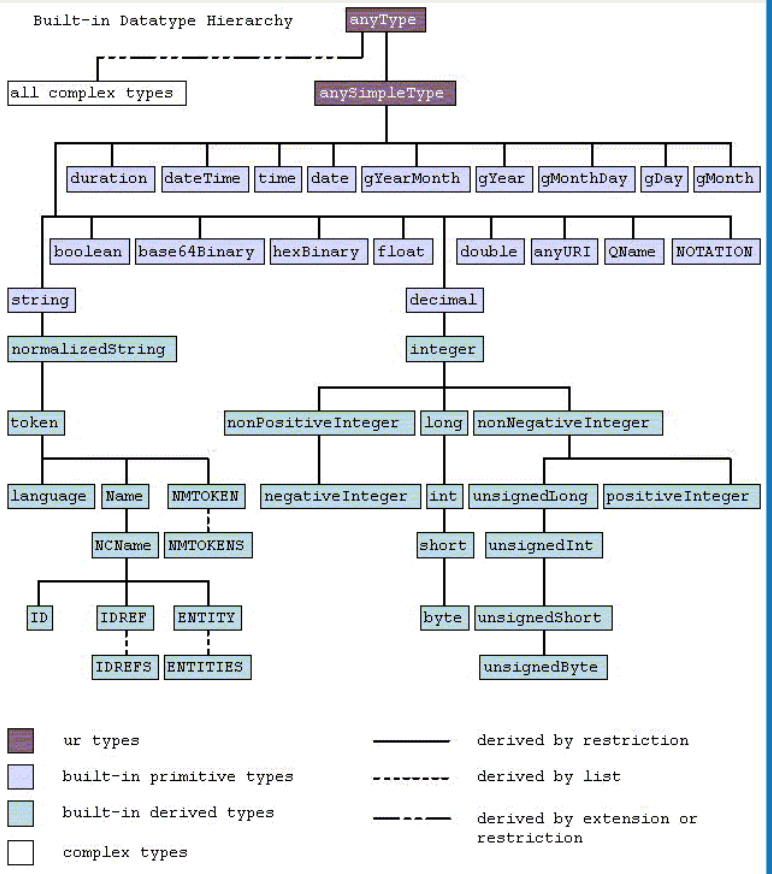
Hiérarchie des types de données (W3C)

Version 1.0 pour commencer.

Source :

https://www.w3.org/TR/xmlschema-2/#datatype

Evolution de la version sera abordée plus tard.



Points abordés

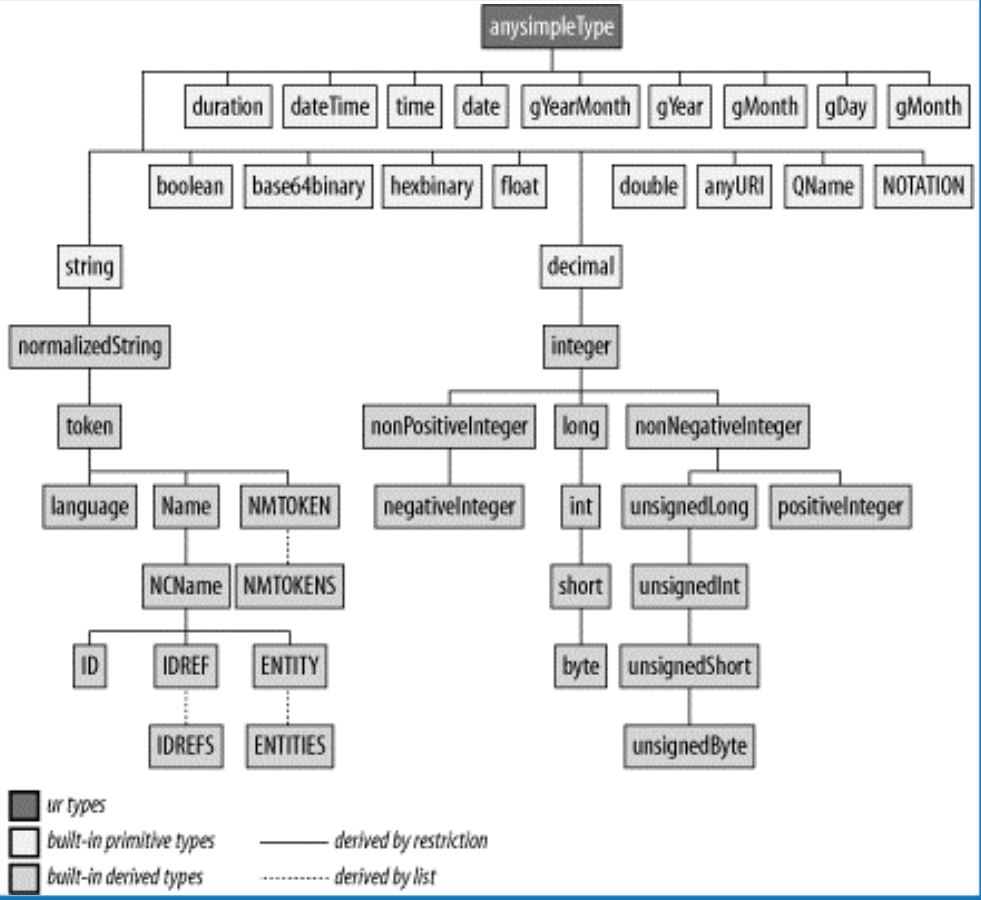
Types prédéfinis pour les données simples (descendants de anySimpleType):

- Types primitifs:
  - Types textes (string, anyURI, …)
  - Types numériques (decimal, float, …)
  - Types temporels (dateTime, date, …)
  - Autres (hexBinary, …)
- Types dérivés par restriction ou par liste:
  - Types textes (normalizedString, token, …)
  - Types numériques (integer, long, …)
  - Types listes (IDREFS, NMTOKENS, …)

Types prédéfinis pour données simples

Source :

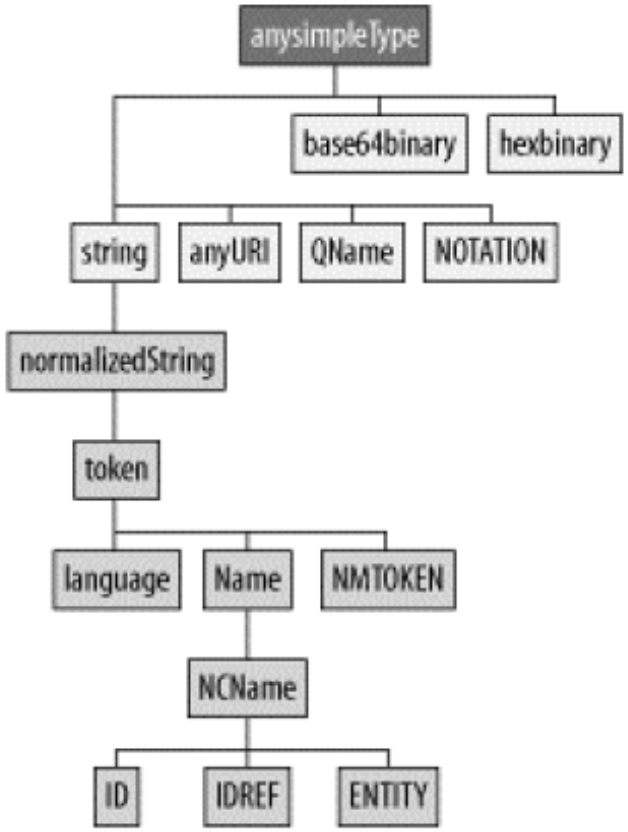
https://www.oreilly.com/library/view/xml-schema/0596002521/ch04.html



Types textes

Source :

https://www.oreilly.com/library/view/ml-schema/0596002521/ch04.html



string

string → type primitif. Il peut contenir tout texte y compris tabulation, retour chariot (carriage return), espaces multiples, …

Exemple de valeurs conformes :

"Ce matin il fait beau

et ce soir il va pleuvoir", …

normalizedString

normalizedString → dérivé de **string**. Pour être conforme, il ne doit pas contenir les caractères suivants :

- retour chariot (carriage return) (#xD)
- saut de ligne (line feed) (#xA)
- tabulation (tab) (#x9)

Exemple de valeurs conformes :

"Ce matin il fait beau et ce soir il va pleuvoir", …

Exemple de valeurs non conformes :

"Ce matin il fait beau et ce soir il va pleuvoir" (tab), …

token

token → dérivé de **normalizedString**. Pour être conforme, il ne doit pas contenir les caractères suivants :

- Pas d'espaces multiples, pas d'espaces avant le début et la fin du texte

Exemple de valeurs conformes :

"Ce matin il fait beau et ce soir il va pleuvoir", …

Exemple de valeurs non conformes :

" Ce matin il fait beau et ce soir il va pleuvoir " (pas d'espaces avant "Ce matin…" et après "…pleuvoir"), …

language

language → dérivé de **token**. Il permet de stocker des codes de langues standardisé selon le RFC 4646 et 4647 (remplace 1766 et 3066).

Exemple de valeurs conformes :

"en": langue anglaise

"fr": langue française

"fr-CH": langue française utilisée en Suisse

…

Ce type a un impact significatif sur le système d'information selon son utilisation.

NMTOKEN

NMTOKEN → dérivé de **token**. Il correspond au Nmtoken de XML 1.0 et la valeur correspond à une seule chaîne de caractères selon le type **token**, mais sans espace.

Exemple de valeurs conformes :

"Monsieur", "Madame" ou "001", "002", …

Exemple de valeurs non conformes :

"Monsieur Madame" (espace interdit), "001,002" (virgule interdite), …

Name

Name → dérivé de **token**. Il correspond au Name de XML 1.0. La valeur doit commencer par une lettre, ":" ou "–".

Exemple de valeurs conformes :

"-oui", ":non", "oui", …

Exemple de valeurs non conformes :

"001" (commence par un chiffre), "oui,non" (virgule interdit), …

NCName

NCName (Noncolonized Name) → dérivé de **Name**. Il correspond aux Namespaces de XML 1.0. Par exemple dans un document XML, le préfixe "xs" de xs:string est de type **NCName**.

Exemple de valeurs conformes :

"-oui", "oui", …

Exemple de valeurs non conformes :

"001" (commence par un chiffre) ou ":non" (commence par deux points), …

ID

ID → dérivé de **NCName**. Il correspond à l'attribut ID de XML 1.0 et la valeur a les mêmes caractéristiques que **NCName**, mais elle doit être unique dans le document XML.

Exemple de valeurs conformes :

"C01", "PROD238", …

Exemple de valeurs non conformes :

"001" ou "12132" (commence par un chiffre), …

IDREF

IDREF → dérivé de **NCName**. Il correspond à l'attribut IDREF de XML 1.0 et la valeur a les mêmes caractéristiques que **NCName**, mais elle doit correspondre à une valeur ID dans le même document XML.

Exemple de valeurs conformes :

"C01", "PROD238", …

Exemple de valeurs non conformes :

"001" ou "12132" (commence par un chiffre), …

ENTITY

ENTITY → dérivé de **NCName.** Ce type de donnée est utilisé à des fins de compatibilité avec un DTD (Document Type Definition) qui est le prédécesseur du XSD (XML Schema Definition).

Il doit correspondre à une entité définie dans un DTD.

Ce type est spécifique pour une utilisation particulière.

Exemple :

https://www.herongyang.com/XSD/string-Datatypes-ENTITY-Values-Representations.html

QName (1)

QName (Qualified Name) est un type primitif et sa valeur est une chaîne de caractères. Il correspond au nom qualifié de XML qui représente un ensemble de tuples.

La valeur de ce type doit respecter le nommage XML et la contrainte est que cette valeur ne peut contenir qu'une seule fois ":" et pas au début.

Ce type a été pensé pour le XML.

QName (2)

Illustration:

{namespace name (préfixe), local part (nom) } →

{xs:attribute, xs:element, xs:string, xs:language, …}

Le préfixe "xs" est associé au namespace (anyURI) qui est défini au début du document XML (XSD).

Le local part "attribute" est de type **NCName**.

"xs"+":"+"attribute" correspond au type **QName**.

anyURI

anyURI est un type primitif et sa valeur est une chaîne de caractères. Cette valeur représente un URI (Uniform Resource Identifier) selon le RFC 2396 et 2732.

Exemple de valeurs conformes :

"https://www.google.com/", …

Exemple de valeurs non conformes :

"Jean@dupond:http" (ce n'est pas un URI correct), …

NOTATION

NOTATION est un type primitif et sa valeur est une chaîne de caractère. Il a été créé pour implémenter les notations de XML 1.0. Ce type fournit un mécanisme permettant à un parseur XML de localiser des programmes externes ou des instructions de traitement.

Ce type est spécifique pour une utilisation particulière.

Exemple :

https://www.oreilly.com/library/view/xml-schema/0596002521/re39.html

hexBinary

hexBinary est un type primitif et il permet de coder du contenu (texte,...) en hexadécimal. Chaque caractère est codé sur deux caractères en format hexadécimal.

Exemple de valeur textuelle :

"<?xml version="1.0" encoding="UTF-8"?>", ...

Exemple de valeurs au format hexBinary :

"3f3c6d78206c657673726f693d6e3122302e20226e656f
636964676e223d54552d4622383e3f", ...

base64Binary

base64Binary est un type primitif et il permet de coder du texte selon le format "base64" (RFC 2045).

Chaque caractère est codé en paquets de 6 bits par caractère.

Exemple de valeur textuelle :

"<?xml version="1.0" encoding="UTF-8"?>", …

Exemple de valeurs au format base64Binary :

"PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4NCg==", …

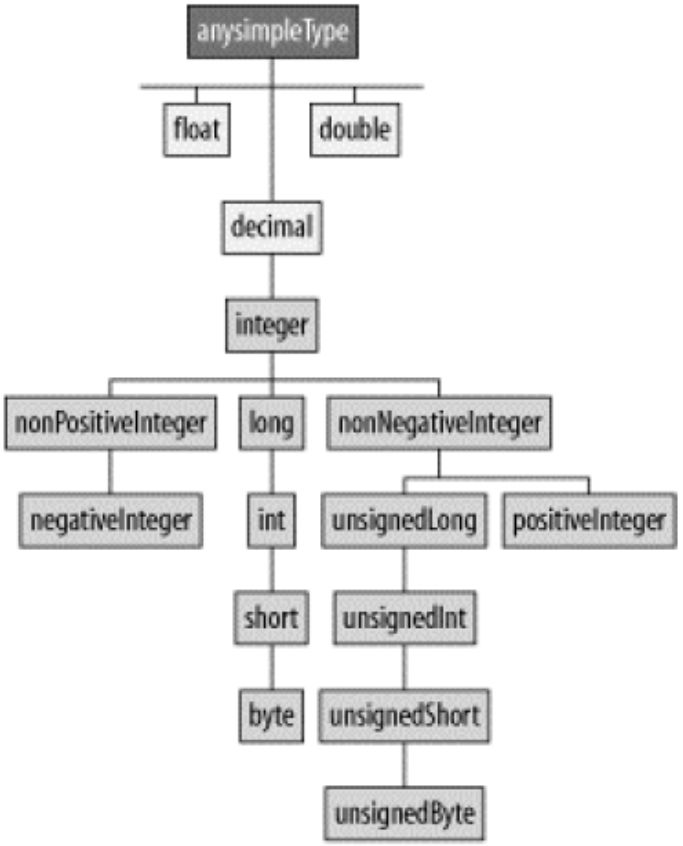
Types numériques (1)

- Construits sur la base de trois types primitifs
- decimal → nombres entiers
- float et double → nombres réels
- Pour être conformes les chiffres ne doivent pas avoir d'espaces

Types numériques (2)

Source :

https://www.oreilly.com/library/view/xml-schema/0596002521/ch04.html



decimal (1)

decimal est un type primitif et permet de saisir des nombres décimaux (base 10).

Le "." est le séparateur entre la partie entière et fractionnaire.

Les chiffres 0 à 9 sont autorisés, ainsi que le signe "+" et "-" devant le nombre.

Tout autre caractère (espace, ...) est interdit.

decimal (2)

La taille du nombre n'est pas limitée par le type.

Exemple de valeurs conformes :

"25.99", "+25.99", "-25.99", "25", "-.99", …

Exemple de valeurs non conformes :

"2.55E+2" (notation scientifique E+2 interdit),

"1'25.99" (séparateur pour les milliers interdit),

"1 25.99" (espace interdit), …

integer

integer → dérivé de **decimal**. Sa valeur n'a pas de partie fractionnaire. La taille du nombre n'est pas limitée par le type.

Exemple de valeurs conformes :

"25", "+25", "-25", …

Exemple de valeurs non conformes :

"1'25" (séparateur pour les milliers interdit),

" 25" (espace interdit), …

Types entiers dérivés (1)

nonPositiveInteger → dérivé de **integer**

: [-∞ ; 0]

negativeInteger → dérivé de **nonPositiveInteger**

: [-∞ ; -1]

nonNegativeInteger → dérivé de **integer**

: [0; ∞]

positiveInteger → dérivé de **nonNegativeInteger**

: [1; ∞]

Types entiers dérivés (2)

long → dérivé de **integer** : [-9223372036854775808 à 9223372036854775807] (64 bits)

int → dérivé de **long** : [-2147483648 à 2147483647] (32 bits)

short → dérivé de **int** : [-32768 à 32767] (16 bits)

byte → dérivé de **short** : [-128 à 127] (8 bits)

Types entiers dérivés (3)

unsignedLong → dérivé de **nonNegativeInteger**
: 0 à 18446744073709551615

unsignedInt → dérivé de **unsignedLong**
: 0 à 4294967295

unsignedShort → dérivé de **unsignedInt**
: 0 à 65535

unsignedByte → dérivé de **unsignedShort**
: 0 à 255

float et double (1)

float et double sont des types primitifs et permettent de saisir des nombres réels simples (32 bits) et double (64 bits).

Exemple de valeurs conformes :

"0", "-0", "INF" (infini positif), "-INF" (infini négatif), "NaN" (Not a Number),"2.55E+2", …

Exemple de valeurs non conformes :

"+INF" (pas de signe positif pour infini),

"2.55 E+2" (espace interdit), …

boolean

boolean est un type primitif et permet de saisir des valeurs logiques (vrai ou faux). Dans ce type la valeur vraie (true) vaut 1 et la valeur fausse (false) vaut 0.

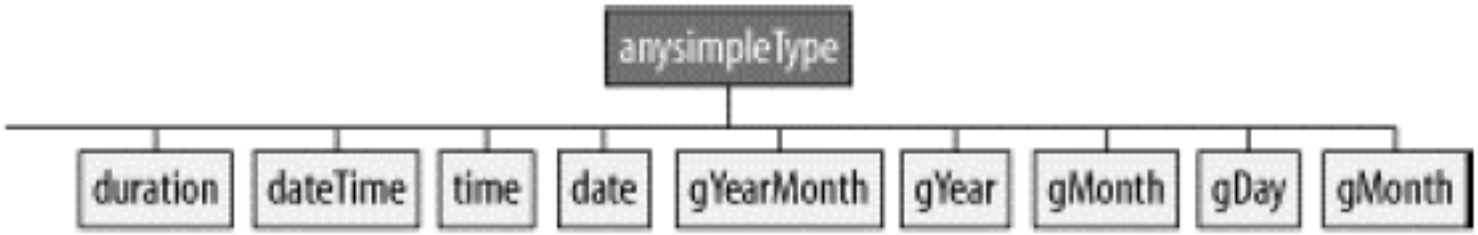
Type boolean n'est pas un type numérique comme decimal, float ou double → Car pas d'opérations arithmétiques possibles (+, *, -, /, …).

Exemple de valeurs conformes :

"true" ou "1", "false" ou "0"

Dates et Heures

Source : https://www.oreilly.com/library/view/xml-schema/0596002521/ch04.html



Dates et Heures

La recommandation W3C utilise le standard ISO 8601 pour éviter tout risque de confusion pour les dates.

Impose l'usage du calendrier Grégorien.

Le Time Zone peut être aussi indiqué de manière optionnelle (+01:00, -01:00 ou Z qui est équivalent à +00:00 ou -00:00).

dateTime

dateTime est un type primitif et permet de saisir une valeur spécifique de temps. Cette valeur doit respecter le format suivant : CCYY-MM-DDThh:mm:ss[Time Zone].

Exemple de valeurs conformes :

"2022-08-15T08:00:52", "2022-08-15T08:00:52Z",

"2022-08-15T08:00:52+02:00", …

Exemple de valeurs non conformes :

"2022-08-15" (manque les données des heures), …

date

date est un type primitif et permet de saisir une partie de la date de **dateTime**.

Exemple de valeurs conformes :

"2022-08-15", "2022-08-15+02:00", …

Exemple de valeurs non conformes :

"2022-08" (manque la donnée du jour),

"2022-13-15+02:00" (13 n'est pas un mois correct), …

gYearMonth

gYearMonth est un type primitif et permet de saisir l'année et le mois d'une date.

Exemple de valeurs conformes :

"2022-08", "2022-08+02:00", …

Exemple de valeurs non conformes :

"2022" (manque la donnée du mois),

"22-08" (format de l'année est incorrect), …

gYear

gYear est un type primitif et permet de saisir uniquement l'année d'une date.

Exemple de valeurs conformes :

"2022", "2022+02:00", …

Exemple de valeurs non conformes :

"2022-08" (le mois n'est pas autorisé),

"22" (format de l'année est incorrect), …

time

time est un type primitif et permet de saisir uniquement la partie temps (hh:mm:ss) de **dateTime**.

Exemple de valeurs conformes :

"08:02:55", "08:02:55+02:00", "08:02:55Z", …

Exemple de valeurs non conformes :

"08:02" (manque les secondes),

"25:02:55" (25 n'est pas une heure correcte),

"8:02:55" (format de l'heure incorrect), …

gDay

gDay est un type primitif et permet de saisir un jour du calendrier Grégorien au format : ---DD[Time Zone].

Exemple de valeurs conformes :

"---01", "---14Z", "---31+02:00", …

Exemple de valeurs non conformes :

"--01-" (ne respecte pas le format de gDay),

"---32" (32 n'est pas un jour correct du calendrier),

"01" (manque "---" devant 01), …

gMonthDay

gMonthDay est un type primitif et permet de saisir un mois et un jour du calendrier Grégorien --MM-DD[Time Zone].

Exemple de valeurs conformes :

"--08-15", " --08-15Z", " --08-15-02:00", …

Exemple de valeurs non conformes :

"-08-15-" (ne respecte pas le format de gMonthDay),

"--08-32" (32 n'est pas un jour correct), …

gMonth

gMonth est un type primitif et permet de saisir un mois du calendrier Grégorien --MM[Time Zone].

Exemple de valeurs conformes :

"--01", " --06Z", "--12-06:00", …

Exemple de valeurs non conformes :

"-08" (ne respect pas le format de gMonth),

"--08-15" (le jour 15 n'est pas autorisé),

"--13" (13 n'est pas un mois correct), …

duration (1)

duration est un type primitif et permet de saisir une durée de temps selon le format "PnYnMnDTnHnMnS" (ISO 8601). Ce format se décrit de la manière suivante :

- P:début (obligatoire), n: valeur de l'année, Y:année, n: valeur du mois, M:mois, n: valeur du jour, D:jour,
- T:début zone temps, n: valeur de l'heure, H:heure, n: valeur de la minute, M:minute, n: valeur de la seconde, S:seconde.

Le problème est la précision de la durée, car :

- 1 mois = entre 28 et 31 jours
- 1 année = 365 ou 366 jours

duration (2)

A prendre en compte également que:

- Les nombres d'années, de mois, de jours, d'heures, de minutes et de secondes sont des entiers positifs.
- Les nombres de secondes peuvent être indiqués au format décimal positif.
- Il est possible d'exprimer une durée négative (-P)

Exemple de valeurs conformes :

"P2Y4M8DT12H24M16S" → durée de 2 ans, 4 mois, 8 jours, 12 heures, 24 minutes et 16 secondes

"-P60Y" → durée de moins de 60 ans, …

duration (3)

Exemple de valeurs non conformes :

"10Y" (manque le P pour le début),

"P-10Y" (nombre négatif interdit à l'intérieur),

"P30S" (il manque le T après le P),

"P6M2Y" (selon format du type duration, l'ordre n'est pas respecté), …

Types Listes (1)

Il existe trois types de listes prédéfinis :

- NMTOKENS → liste de NMTOKEN
- IDREFS →liste de IDREF
- ENTITIES →liste de ENTITY

Ces trois types de listes sont des types dérivés.

Les éléments de la liste sont séparés par un espace blanc.

Chaque élément de la liste doit respecter les contraintes de son type dérivé (IDREF, …).

Types Listes (2)

Exemple de valeurs conformes pour le type IDREFS :

"C01 C02 C03 C04", "PROD1 PROD2", …

Exemple de valeurs non conformes pour le type IDREFS :

"1 2 3 4" (ne respecte pas la contrainte du type IDREF), …

anySimpleType

anySimpleType est le type de base de tout type simple primitif et accepte n'importe quelle valeur sans aucune restriction. Ce type ne peut pas être dérivé et doit être utilisé uniquement quand on ne peut pas faire autrement.

Exemple de valeurs conformes :

"--01", " 128.331", " 2022-08-15T08:00:52", "Le petit train", …

En conclusion

- Différents types de données prédéfinis existent.
- Un certain nombre de types préfinis (numériques, textes, dates, …) permettent d'améliorer la gestion des données et de leur précision (format, limite, …).
- Certains types de données sont plus spécifiques à des domaines particuliers (QName, NCName, ENTITY, …).
- Ces types de données contribuent à améliorer la qualité des données d'un document XML.
- …

Références

- XML Schema Part 2 (W3C): https://www.w3.org/TR/xmlschema-2/#dt-anySimpleType
- "Chapter 4. Using Predefined Simple Datatypes" du livre "XML Schema" en ligne: https://www.oreilly.com/library/view/xml-schema/0596002521/ch04.html
- Mon expérience dans le domaine, …
- Livre "XML en concentré" →

Merci de votre attention !