



**Séance 6**

# **Xpath**

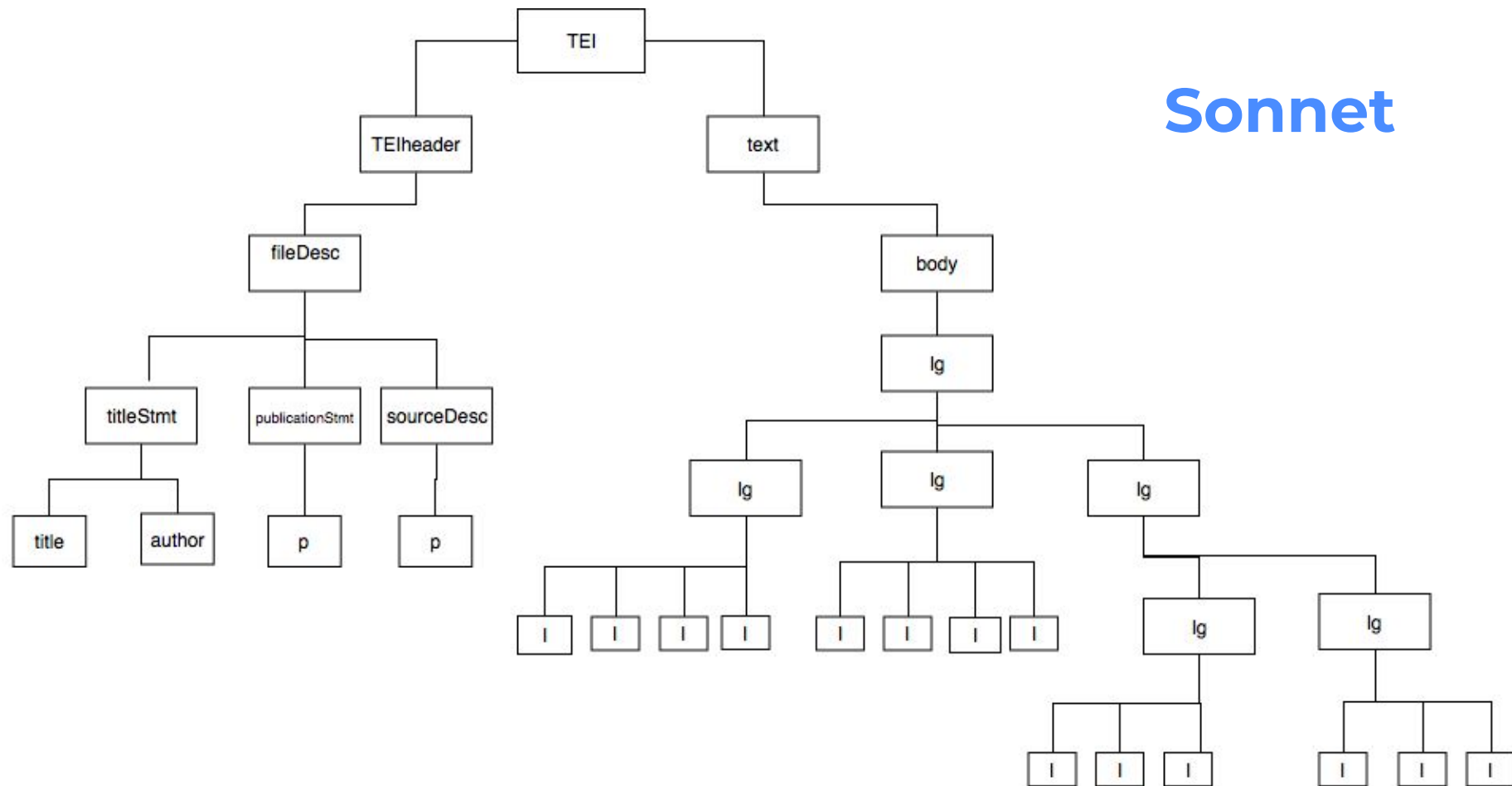
Localisation dans le document XML

A decorative graphic on the left side of the slide consisting of two overlapping squares. The bottom-left square is a dark blue, and the top-right square is a lighter blue, creating a cross-like shape.

# Structure en arbre

Maîtriser la hiérarchie XML

# Sonnet



# Sélectionner des nœuds

Le langage Xpath fournit une syntaxe pour sélectionner un nœud ou un ensemble de nœuds au sein d'un document XML.

# Tester son Xpath

Oxygen permet de surligner les nœuds sélectionnés par un chemin avec la **barre Xpath** + **Entrée**



```
xt  body  lg  lg  l
```

```
<lg type="sonnet">
  <head>Sonnet d'automne</head>
  <lg type="quatrain" n="1">
    <l>Ils me disent, tes yeux, clairs comme le cristal :</l>
    <l>« Pour toi, bizarre amant, quel est donc mon mérite ? »</l>
    <l>— Sois charmante et tais-toi ! Mon cœur, que tout irrite,</l>
```

A decorative graphic on the left side of the slide consisting of two overlapping squares. The bottom-left square is a dark blue, and the top-right square is a lighter blue, creating a cross-like shape.

# Qu'est-ce que le Xpath ?

Les principes du langage

# Principes Xpath

## W3C

Xpath 2.0 est publié par le *World Wide Web Consortium* comme standard XML

## Requêtage

Xpath est un **langage de requêtes** qui permet de parcourir un arbre XML

## Intégré

Xpath a été conçu comme un **langage intégré**, et non pas un langage autonome

# Syntaxe chemin

```
node/childNode
```

Une expression de chemin correspond à une **séquence d'étapes** séparées par l'opérateur `/`.

Sans indication particulière la relation se fait d'un élément parent vers un élément enfant.

Chaque étape devient le nœud courant (nœud de contexte) pour l'étape suivante.



# Absolu / relatif

Un chemin de localisation **absolu** commence au nœud document : **il commence par une barre oblique /**

Les chemins de localisation **relatifs** commencent à un nœud de l'arborescence. Ils sont utilisés notamment au sein d'un autre chemin Xpath.

# Désignation d'une localisation

Une expression XPath définit un chemin de localisation, constitué de **pas de localisation** ("étape du chemin") séparés par le caractère `/`.

Les pas sont définis par :

## Axe

**Sens** dans lequel l'arbre doit être parcouru  
parcoursu (`parent::`,  
`descendant::`)

## Test de nœud

**Nom** d'un nœud ou  
**type** de nœud situé  
dans la direction de l'axe  
dans l'arborescence

## Prédicat(s)

Indiqué entre `[ ... ]`  
pour préciser une  
caractéristique du nœud  
(sorte de **filtre**)

# Syntaxe pas de localisation

```
axe::nœud[predicat1][predicat2]...
```

A decorative graphic on the left side of the slide consisting of two overlapping squares. The bottom-left square is a dark blue, and the top-right square is a lighter blue, creating a cross-like shape.

# Syntaxe Xpath

Les différents éléments manipulés en Xpath



# Les nœuds

Les documents XML sont traités comme des **arbres de nœuds**. Les nœuds désignent des composants et contenus du document XML.

L'ordre des nœuds est déterminé par la séquence du document et par l'imbrication des éléments XML.

# Types de nœuds courants

`elementName`    ***élément***

Désigne un élément (balises et contenu)  
Ne concerne pas les déclarations (<?...?> et <!--...-->)

`@attributeName`    ***attribut***

Désigne l'attribut d'un élément XML

`text()`    ***texte***

Désigne le contenu textuel **directement** contenu  
dans un élément (sans la balise)

`comment()`    ***commentaire***

Désigne le contenu d'un commentaire (pas son  
balisage <!--...-->)

# Un élément et son contenu

```
/TEI/teiHeader/titleStmt/title
```

L'élément `title` et son contenu

E.g.: `<title type="play">Le Misanthrope</title>`

# Le texte d'un élément

```
/TEI/teiHeader/titleStmt/title/text()
```

La chaîne de caractère contenu dans l'élément `title`

E.g.: “Le Misanthrope”



# La valeur d'un attribut

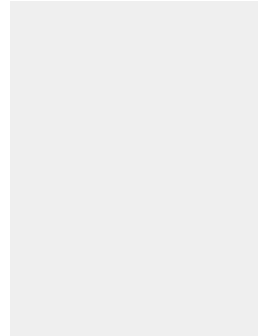
```
/TEI/teiHeader/titleStmt/title/@type
```

La valeur de l'attribut `type` de l'élément `title`

E.g.: "play"

# Exercice

**Sélectionner les nœuds**



# Consigne

Trouver les chemins Xpath dans l'encodage [TEI d'un sonnet](#) :

Les vers de quatrain

Le paragraphe de provenance

Les tercets

Le titre du poème

# Consigne

Trouver les chemins Xpath dans l'encodage **TEI d'un sonnet** :

Les vers de quatrain

`/TEI/text/body/lg/lg/l`

Le paragraphe de provenance

`/TEI/teiHeader/fileDesc/sourceDesc/p`

Les tercets

`/TEI/text/body/lg/lg/lg`

Le titre du poème

`/TEI/text/body/lg/head`

# Abréviations pour les nœuds

- \* Remplace n'importe quel nœud
- | Pour combiner deux ensembles de nœuds

# N'importe quel contenu

```
/TEI/teiHeader/*
```

Tout ce qui est contenu dans le `teiHeader`

# N'importe quelle étape

```
/TEI/teiHeader/*/div
```

Toute `div` contenue dans un sous-élément du `teiHeader`

# Concaténation de chemins

```
/TEI/teiHeader | /TEI/text/body
```

Le `teiHeader` et le `body`

⚠ L'ordre des chemins doit suivre l'ordre d'apparition dans le document. Par exemple, `/TEI/text | /TEI/teiHeader` ne sélectionne que le `text`.



# Concaténation une étape

```
/TEI/(teiHeader/fileDesc|text/body)/.../p
```

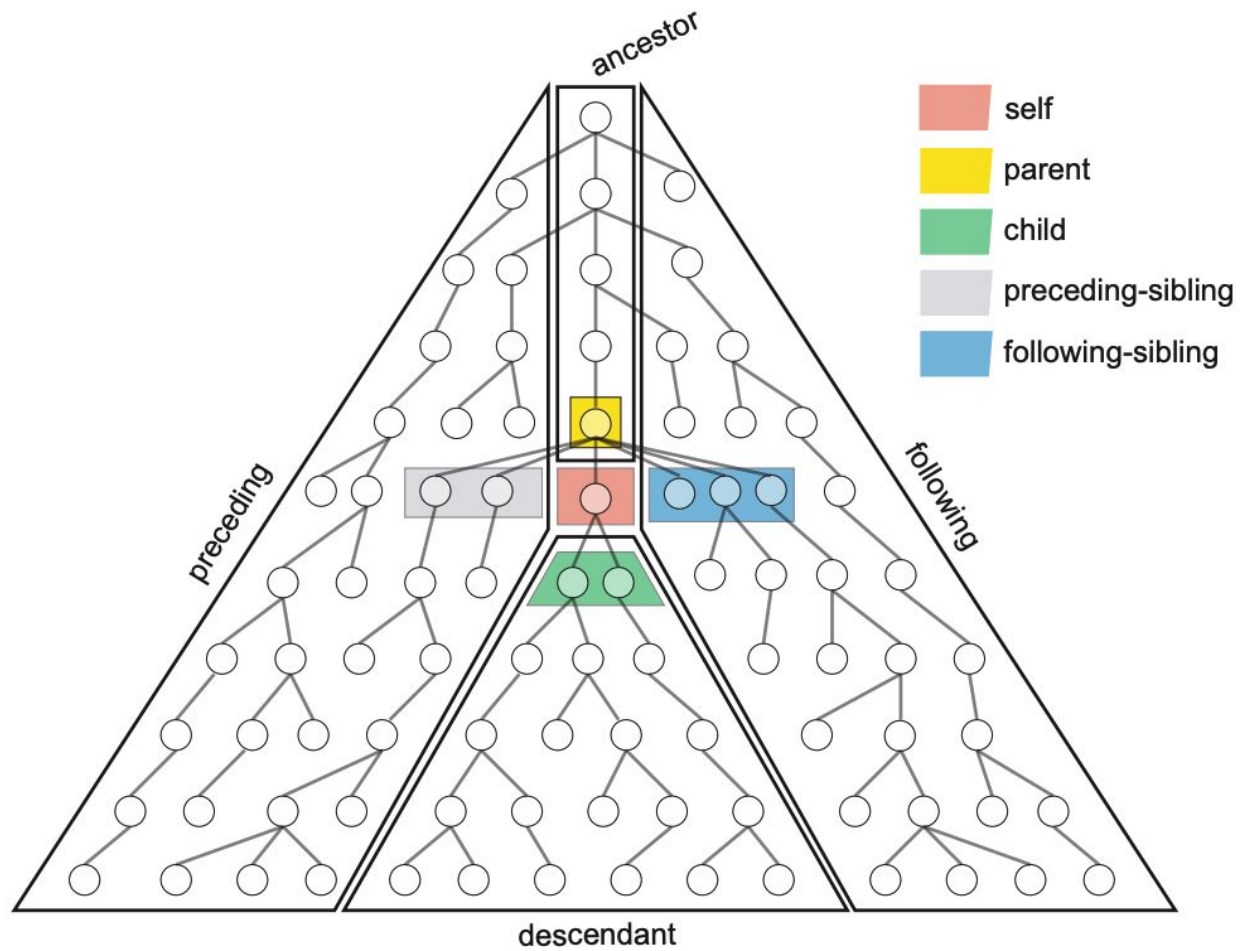
Les paragraphes contenus dans le `fileDesc` et le `body`

# Axes Xpath

Indique la **direction** dans laquelle se déplacer dans l'arbre XML, relativement au nœud courant ou depuis la racine. Quand l'axe n'est pas précisé, il s'agit implicitement de l'axe des enfants (`child::`).

# Axes courants

<b>ancestor</b>	Sélectionne tous les ancêtres du nœud courant
<b>ancestor-or-self</b>	Sélectionne tous les ancêtres du nœud courant ainsi que le nœud lui-même
<b>attribute</b>	Sélectionne tous les attributs du nœud courant
<b>child</b>	Sélectionne tous les enfants du nœud courant
<b>descendant</b>	Sélectionne tous les descendants du nœud courant
<b>descendant-or-self</b>	Sélectionne tous les descendants du nœud courant ainsi que le nœud lui-même
<b>following</b>	Sélectionne tous les nœuds du document après la balise fermante du nœud courant
<b>following-sibling</b>	Sélectionne tous les nœuds suivants qui sont au même niveau que le nœud courant
<b>parent</b>	Sélectionne le nœud parent du nœud courant
<b>preceding-sibling</b>	Sélectionne tous les nœuds précédents qui sont au même niveau que le nœud courant
<b>self</b>	Sélectionne le nœud courant



# Syntaxe axe attribute

```
/TEI/.../p/attribute::n
```

La valeur de l'attribut n des éléments p contenus dans le document

# Syntaxe axe parent

```
/TEI/.../p/parent::*
```

Les éléments pour contiennent directement un p

# Syntaxe axe sibling

```
/TEI/.../lg/  
(preceding-sibling::*|following-sibling::*)
```

Les éléments au même niveau que les lg du document

# Les abréviations d'axe

<b>vide</b>	<code>child::</code>	Axe par défaut
<b>@</b>	<code>attribute::</code>	Sélectionne la valeur d'un attribut
<b>//</b>	<code>/descendant::node()/</code>	Sélectionne parmi les descendants
<b>..</b>	<code>parent::node()</code>	Sélectionne le nœud parent
<b>.</b>	<code>self::node()</code>	Sélectionne le nœud courant



# Optimisation

Il est préférable d'éviter d'utiliser `//` en début de chemin pour des questions de performance et de non ambiguïté

```
/TEI//p
```

# N'importe où dans l'arborescence

```
//p
```

Tous les éléments p peu importe leur endroit dans l'arborescence

# Sélection d'attribut

//p/@n

La valeur des attributs n des éléments p

# Parmi les descendants

```
//div//p
```

Tous les éléments `p` descendants d'un élément `div` dans le document

# Parmi les descendants

`./p`

Tous les éléments p descendant du nœud courant

# Noeud parent

//p/ . .

Les éléments contenant directement un élément p

# Noeud parent spécifique

```
//p/parent::div
```

Les div contenant directement un élément p

# Contenu textuel

```
//lg//text()
```

Le texte contenu dans les descendants de tous les éléments lg



# Exercice

**Traduire les expressions**

# Consigne

Traduire les expressions suivantes

```
//figure
```

```
//*/*
```

```
//book/title
```

```
chapter//footnote
```

```
./footnote
```

# Consigne

Traduire les expressions suivantes

`//figure`

`//*/*`

`//book/title`

`chapter//footnote`

`./footnote`

N'importe quels éléments **figure** dans le XML

N'importe quel nœud contenu dans un autre

Tous les **title** contenu **directement** dans **book**

Toutes les **footnote** contenu dans un **chapter**

Toutes les **footnote** contenu dans le nœud courant



# Prédicats

Les prédicats sont utilisés pour **filtrer** les nœuds sélectionnés par l'axe et le test de nœud. Les prédicats sont écrits entre crochets. Si le prédicat est évalué à vrai, tous les nœuds correspondants sont sélectionnés.


## Noeud qui contient...

```
//title[.="Sonnet d'automne"]
```

Les éléments `title` qui ont pour contenu "Sonnet d'automne"

Cette syntaxe abrégée est l'équivalent de :

```
[text()="Sonnet d'automne"]
```

 Attention, un prédicat avec seulement du texte entre guillemets ne filtre rien `["Sonnet d'automne"]`

## Noeud qui a un enfant qui...

```
//titleStmt[title="Sonnet d'automne"]
```

Les éléments `titleStmt` dont l'enfant `title` a pour contenu "Sonnet d'automne"

## Noeud qui a un enfant qui...

```
//titleStmt[title="Sonnet d'automne"]/author
```

Les éléments `author` contenus dans les éléments `titleStmt` dont les enfants `title` ont pour contenu "Sonnet d'automne"

## Noeud qui a un descendant qui...

```
//div[.//persName="Cendrillon"]
```

Les éléments `div` qui contiennent un élément `persName` a pour contenu "Cendrillon"



# Test d'existence

```
//titleStmt[title]
```

Les éléments `titleStmt` qui contiennent un élément `title`

# Élément qui a pour attribut...

```
lg[@type]
```

Les éléments lg qui ont un attribut type

## Élément qui a pour attribut...

```
lg[@type="quatrain"]
```

Les éléments lg qui ont un attribut type ayant pour valeur quatrain

# Prédicat numérique

```
lg[@type="quatrain"][2]
```

Le deuxième élément `lg` qui a un attribut `type` ayant pour valeur `quatrain`

Abréviation de `lg[position()=2]`

 Contrairement aux autres langages de programmation, l'énumération XPath commence avec 1

# Exercice

**Filtrer les nœuds**



# Consigne

Traduire ces expressions en xPath

Le second `p` de la première `div` du `body`

Les `p` au sein du `text` qui contiennent des `persName`

Les `placeName` dont la valeur est Paris

La troisième mention du `persName` "Gringoire"

La deuxième `div` contenue directement ou non dans la quatrième `div`

# Consigne

Traduire ces expressions en xPath

Le second p de la première div du body

```
//body/div[1]/p[2]
```

Les p au sein du text qui contiennent des persName

```
//text//p[persName]
```

Les placeName dont la valeur est Paris

```
//placeName[.="Paris"]
```

La troisième mention du persName "Gringoire"

```
//persName[.="Gringoire"][3]
```

La deuxième div contenue directement ou non dans la quatrième div

```
//div[4]//div[2]
```

# Opérateurs Xpath

Les tests formulés dans les prédicats sont définis à l'aide d'opérateurs. Il en existe de trois types : les opérateurs **arithmétiques**, les opérateurs **logiques** et les opérateurs de **comparaison**.



# Opérateurs courants

<code>+</code> <code>-</code> <code>*</code> <code>div</code> <code>mod</code>	Pour effectuer des opérations arithmétiques
<code>=</code> <code>!=</code>	Pour tester l'égalité ou l'inégalité
<code>&lt;</code> <code>&gt;</code>	Strictement plus grand / plus petit que ( <code>&lt;</code> et <code>&gt;</code> en XSLT)
<code>&lt;=</code> <code>&gt;=</code>	Inférieur / supérieur ou égal
<code>and</code> <code>or</code>	Pour accoler plusieurs expressions booléennes

# Comparer des valeurs

```
1[@n > 2]
```

Les éléments 1 dont la valeur de l'attribut n est strictement supérieure à 2

# Combiner des expressions

```
lg[@type != "tercet" and @n < 2]
```

Les éléments lg dont la valeur de l'attribut n est strictement inférieure à 2 et dont l'attribut type n'est pas "tercet"

Équivalent à `lg[@type != "tercet"][@n < 2]`

# Fonctions Xpath

Les fonctions Xpath, utilisées dans les prédicats, permettent de manipuler les données à tester. Elles peuvent s'appliquer à des données **textuelles**, **numériques** et **booléennes**. Elles peuvent également s'appliquer à des **ensembles de nœuds** (qui seront exprimés sous forme de chemin Xpath)

# Fonctions courantes

<code>count(node-set)</code>	Compte le nombre d'occurrences dans un ensemble de nœud
<code>sum(node-set)</code>	Somme des valeurs de type nombre dans un ensemble de nœud (qui ne contient <b>que des valeurs numériques</b> )
<code>true(), false(), not(bool)</code>	Vrai / Faux / Valeur inverse du booléen
<code>string(val), number(val)</code>	Pour changer le type d'une valeur
<code>concat(str, str)</code>	Concaténation de plusieurs chaînes de caractère
<code>start-with(str1, str2)</code>	Renvoie vrai si <code>str1</code> commence par <code>str2</code>
<code>contains(str1, str2)</code>	Renvoie vrai si <code>str1</code> contient <code>str2</code>
<code>string-length(str)</code>	Renvoie la longueur d'une chaîne de caractère
<code>normalize-space(str)</code>	Retire les espaces en début et fin de chaîne, ainsi que les doubles espaces
<code>last()</code>	Dernier nœud du document correspondant au prédicat
<code>position()</code>	Nombre désignant la position d'un élément

# Node-set

Une variable de type node-set est définie par un chemin absolu ou relatif par rapport au noeud courant

```
/TEI//lg[@n = count(1)]
```

*Compte le nombre de 1 contenus dans lg*

```
/TEI//lg[@n = count(/TEI//1)]
```

*Compte le nombre de tous les 1 du XML*

# Filtrer la position

```
//lg[position() != 1]
```

Les éléments lg qui ne sont pas le premier dans l'axe du document

# Additionner des valeurs numériques

```
//div[@price = sum(item)]
```

Les éléments `div` dont la valeur d'attribut `price` correspond à l'addition des valeurs numériques contenues dans les éléments `item` directement contenus dans `div`



# Additionner des valeurs numériques

```
//div[@price = sum(//item)]
```

Les éléments `div` dont la valeur d'attribut `price` correspond à l'addition des valeurs numériques contenues dans les éléments `item` du document XML

# Additionner des valeurs numériques

```
//div[@price = sum(../item)]
```

Les éléments `div` dont la valeur d'attribut `price` correspond à l'addition des valeurs numériques contenues dans les éléments `item` descendant dudit élément `div`

# Histoire de vous embrouiller...

```
//div[sum(//@n)] ⇒ //div[6]
```

L'élément `div` qui a pour position dans le document le même nombre que la somme de toutes les valeurs d'attribut `n` du document

```
//div[.=sum(//@n)] ⇒ //div[.="6"]
```

L'élément `div` qui a pour contenu `text()` le même nombre correspondant à la somme de toutes les valeurs d'attribut `n` du document

# Exercice

**Manipuler les données**



# Consigne

Traduire les formules suivantes

```
//lg[@n = count(l)]
```

```
//l[last()]
```

```
//persName[.="Trucmuche"]
```

```
//persName[contains(., "Truc")]
```

```
//listPerson[person[contains(., "Truc")]]
```

# Consigne

Traduire les formules suivantes

```
//lg[@n = count(l)]
```

Les lg dont le n° est égal au nombre de vers qu'ils contiennent directement

```
//l[last()]
```

Le dernier vers du document

```
//persName[.="Trucmuche"]
```

Les persName dont la valeur est "Trucmuche"

```
//persName[contains(., "Truc")]
```

Les persName dont la valeur textuelle contient la chaîne "Truc"

```
//listPerson[person[contains(., "Truc")]]
```

Les listPerson qui ont pour enfant person qui contient la chaîne "Truc"

# Exercice

A decorative graphic on the right side of the slide, consisting of a light gray square positioned above a blue square, both of which are partially cut off by the right edge of the frame.

**Parcourir les encodages**

# Consigne

Donner les chemins **les plus courts**

## Sonnet d'automne

De la racine vers les vers d'un quatrain

De la racine vers le titre

## Le Misanthrope

De la racine vers les vers d'une scène

De la racine vers la déclaration d'un personnage

De la racine vers la dernière déclaration de personnage dans la liste de la préface



# Consigne

Donner les chemins **les plus courts**

## Sonnet d'automne

```
/TEI//lg[@type="quatrain"]/1
```

```
/TEI//title
```

## Le Misanthrope

```
/TEI//sp/1
```

```
/TEI//speaker
```

```
/TEI//castItem[last()]
```

# Exercice

**Aller plus loin**



# Consigne

Traduire en Xpath les expressions suivantes

Tous les attributs du dernier élément div du body

Tous les nœuds contenant un élément persName dont le contenu est "Molière"

Le parent du dernier persName du teiHeader

L'avant dernier title qui contient la chaîne de caractère "Paris"

Les div parmi les ancêtres des éléments ayant un attribut type

Les div qui contiennent la chaîne de caractère "coucou" quelque part dans leurs descendants

# Consigne

Tous les attributs du dernier élément `div` du `body`

```
TEI//body//div[last()]/@*
```

# Consigne

Tous les nœuds contenant un élément `persName` dont le contenu est "Molière"

```
//*[persName[.="Molière"]]
```

# Consigne

Le parent du dernier persName du teiHeader

```
/TEI/teiHeader//persname[last()]/parent::*
```

# Consigne

L'avant dernier `title` qui contient la chaîne de caractère "Paris"

```
/TEI//title[last()-1][.="Paris"]
```

```
/TEI//title[count(.)-1][contains(., "Paris")]
```

# Consigne

Les div parmi les ancêtres des éléments ayant un attribut type

```
/TEI//*[@type]/ancestor::div
```



# Consigne

Les div qui contiennent la chaîne de caractère "coucou" quelque part dans leurs descendants

```
/TEI//div[.//text()[contains(., "coucou")]]
```

```
/TEI//div[contains(.//text(), "coucou")]
```