Technique et chaîne de publication électronique avec XSLT (7/7)

Jean-Damien Généro

2025, 1er dec. - 2026, 20 janv.

École nationale des chartes – M2 TNAH

Python et XSLT

Python/ Chaîne de publication électronique

- Une chaîne reproductible + documentée + automatisable
- Charger plusieurs données d'entrée (*XML*, *CSV*, *JSON*, *YAML*, *SQL*)
- Manipuler les données (nettoyage, fusion, extraction)
- Produire plusieurs formats de données de sortie adaptés aux intervenants du projet
- Pipeline : charger $\rightarrow$ pré-traiter $\rightarrow$ transformer $\rightarrow$ post-traiter $\rightarrow$ publier

Python/ Un orchestrateur de transformations

- Gestion des fichiers (*`Pathlib`*, *`os`*)
- Chargement des données / data parsing (*`pandas`*, *`json`*, *`lxml`*, *`yaml`*)
- Manipulation des données (variables, *`dict`*, *`list`*, *`dataframe`*, *`xml trees`*)
- Écriture des formats de sortie (chaîne multi-sorties : HTML + LaTeX + PDF)
- Publication des données (API).

- Sortir d'Oxygen (logiciel propriétaire)
- Construire une chaîne de publication électronique cohérente
- XSLT : personnalisation plus fine que d'autres librairies clefs en main (*Pandoc*) : besoins d'un travail scientifique ou patrimonial
- Python ne remplace pas XSLT mais le complète avec des traitements pré-transformation et post-transformation.

Python/ lxml

- Manipulation XML/HTML : parsing, validation, transformation
- Fonctionne avec un fichier ou une string
- Représente le document comme un arbre de nœuds (*`ElementTree`*)
- Implémentation de XPath 1 et XSLT 1

Python/ Chargement d'un XML avec `lxml`

```
from lxml import etree
# charger un fichier: deux solutions
tree = etree.parse("file.xml")
# ou
with open(file, 'r') as xml_file:
    tree = etree.parse(xml_file)
# charger une string
xml_string = "<div><p>Text</p></div>"
tree = etree.fromstring(xml_string)
```

```
from lxml import etree

namespaces = {'tei':
    'http://www.tei-c.org/ns/1.0'}

header = tree.xpath(
    '/tei:TEI/tei:teiHeader',
    namespaces=namespaces)

header[0].attrib['type'] = 'metadata'
```

- NB : la fonction *.xpath()* retourne une liste.
- NB : *attrib* est un *dict*.

Python/ Transformation XSLT

```
from lxml import etree

def xsl_transformation(xsl_file, xml_file):

    source = etree.parse(xml_file)

    xsl_doc = etree.parse(xsl_file)

    xsl_transformer = etree.XSLT(xsl_doc)

    output_doc = xsl_transformer(source)

    return output_doc
```

```
from lxml import etree

str(output_doc).write('output.html',
    encoding='utf-8'
)
```

- *`etree.XSLT()`* n'accepte que XSLT 1
- Solution de contournement :
  - XSLT produit plusieurs arbres XML dans un même fichier
  - Utiliser *`lxml`* pour parser chaque arbre
  - Utiliser python pour les enregistrer un à un