

# RA5: XSL

(extensible stylesheet language)

**Conversió i adaptació de documents XML**

# Conversió i adaptació de documents XML Introducció

XML està pensat sobretot per a emmagatzemar i intercanviar informació, de manera que si cal representar les dades d'una manera diferent per **optimitzar un procés** o per **millorar-ne la visualització** hi haurà diverses possibilitats:

- **Desenvolupar un programa:** com que és relativament senzill treballar amb XML, es podria desenvolupar un programa que agafi les dades XML i generi la sortida tal com la volem. Això té l'inconvenient que caldrà tenir coneixements de programació i que pot representar molta feina encara que el que calgui fer sigui trivial.
- **Fer servir CSS:** en molts casos una **solució senzilla** seria fer servir CSS per representar la informació de manera més amigable fent servir un navegador. Només serveix per canviar la visualització, no per canviar o transformar el document.
- **Transformar el document:** una solució alternativa consisteix a transformar el document en un altre que estigui pensat per ser visualitzat. Hi ha molt formats que estan pensats sobretot per ser visualitzats: PDF, HTML, XHTML, etc.

# Ús de CSS

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <?xml-stylesheet href="estil.css" type="text/css"?>
3 <professorat>
4   <nom>Jordi</nom>
5   <cognom>Garcia</cognom>
6   <departament>Dept. Informàtica</departament>
7   <carrecs>
8     <carrec>Tutor</carrec>
9     <carrec>Docent</carrec>
10    <carrec>Trecent</carrec>
11  </carrecs>
12 </professorat>
```

Jordi Garcia

**Dept. Informàtica**  
*Tutor, Docent, Trecent,*

## Document estil.css

```
professorat{
  padding:30px;
  margin: 30px;
  border: 4px dotted blue;
  width: 40%;
}
nom,cognom{
  font-size: 30px;
}
departament{
  padding-top: 20px;
  display: block;
  font-weight: bold;
}
carrec{
  font-style: italic;
  padding-left: 10px;
}
carrec::after{content: ",";}
```

# Exercici1 css-xml

Crea el fitxer XML i el fitxer CSS i aconseguix que tingui el següent aspecte:

Meritxell Durany

**Dept. Informàtica**

*M04, M2, M8,*

# Ús CSS

Encara que visualment li hem donat un bonic format, no obstant això:

- ❑ La informació **no** pot ser **reordenada**. → L'única manera seria refer el XML.
- ❑ **No** podem **afegir** noves estructures o elements (ex. Càlculs). No es pot modificar el contingut
- ❑ És complicat fer un format per a ser directament imprimible, hem de passar per un navegador.

## CSS no ens serveix:

Si el nostre objectiu és no sols decorar el nostre fitxer **XML**, sinó transformar-lo en un altre XML,

# Transformació de documents XML

- ❑ XSL
- ❑ XSLT
- ❑ XPath

# Transformació de documents XSL

## (eXtensible Stylesheet Language)

Per intentar aconseguir fer tot allò que CSS no podia fer es va crear un nou llenguatge de plantilles: XSL

XSL és una família de llenguatges que serveixen per definir transformacions i presentacions de documents XML

Està formada per aquests 3 llenguatges:

- XSL-FO (XSL formatting objects): un llenguatge per definir el format que s'ha d'aplicar a un document.
  - Sobretot es fa servir per generar documents en formats com PDF o PostScript ( per ser impresos)
- XPath un llenguatge per accedir a parts dels documents XML
- XSLT (XSL transformations): un llenguatge per **transformar** documents XML.

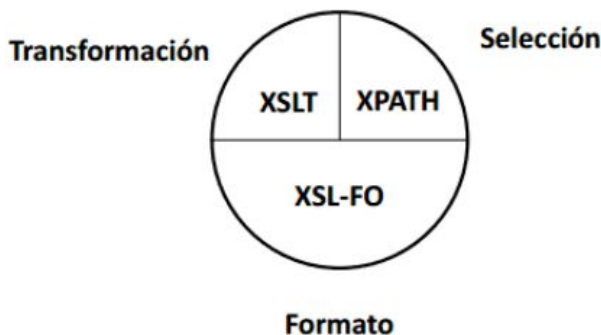


# Processadors XPath i XSLT

En general els processadors XPath o XSLT es proporcionen per mitjà de biblioteques que podran ser cridades des dels programes.

**XPath** (XML path language) és una manera d'especificar parts d'un document XML que té eines per manipular el contingut de les dades de text, numèriques, etc.

**XSLT** (extensible **stylesheet language** for **transformations**) és un llenguatge de plantilles basat en XML que permet convertir l'estructura dels elements XML en altres documents.





# Transformació de documents XML

- ❏ XSL
- ❏ XSLT
- ❏ XPath

# XSLT (eXtensible **S**tylesheet **L**anguage **T**ransform)

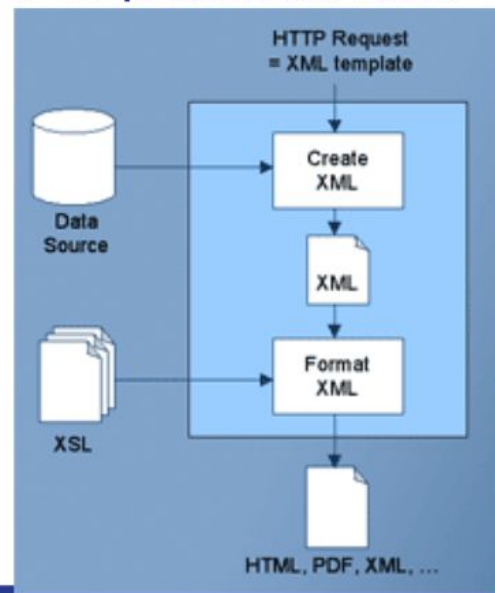
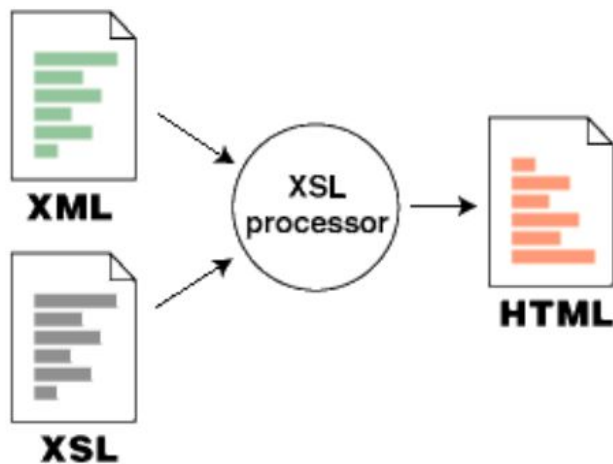
- ❑ Es fa servir per a transformar un document XML en un altre document **XML, HTML**, etc.
- ❑ XSLT pot **afegir** nous elements o fins i tot **eliminar-los**. També **fer proves** o **prendre decisions**.
- ❑ Pot ser utilitzat per transformar documents XML en HTML **abans de ser mostrats** al navegador.

# XSLT (eXtensible **S**tylesheet **L**anguage **T**ransform)

La principal característica del llenguatge **XSLT** és la seva potència.

No és només un llenguatge per a visualitzar documents, sinó en general per a **transformar-los** i **manipular-los**.

Aquesta manipulació la gestiona un programa especial que es diu **processador XSLT**.



## Estructura d'un document XSLT (eXtensible Stylesheet Language Transform)

- ❑ Un full d'estil XSLT és un document XML i, per tant, ha d'estar **ben format**
- ❑ Els fulls XSLT es guarden en arxius independents amb extensió **\*.xsl**
- ❑ Han de començar amb una declaració XML `<?version="1.0"?>`
- ❑ L'element arrel de XSLT és `<xsl:stylesheet>`
- ❑ Aquest element conté la resta d'elements i ha d'anar precedit per l'àlies XSL corresponent a l'espai de noms per a la fulla XSLT.
- ❑ Entre les marques d'inici i final `<...>` de l'element arrel `xsl:stylesheet`, s'escriuran les regles de transformació, que s'anomenen plantilles
- ❑ Cada regla es definirà mitjançant l'element `<xsl:template>`
- ❑ L'atribut **match** s'utilitza per associar cada plantilla amb un element XML, el valor de l'atribut és una expressió XPath
- ❑ La regla indica **quines instàncies** dels elements del document XML es transformaran

**ejemplo**

```
<xsl:template match="//nombre">
  <html>
    <body>
      <h2> <xsl:value-of select="." /> </h2>
    </body>
  </html>
</xsl:template>
```

árbol de origen al que se aplica esta plantilla

salida

# XSLT (eXtensible Stylesheet Language Transform)



Origen:

```
<documento>
  <parrafo>
    mi primer parrafo
  </parrafo>
</documento>
```

XML

Resultado:

```
<html>
<body>
  <p>mi primer parrafo</p>
</body>
</html>
```

HTML

Se vería:

mi primer parrafo

## Estructura d'un document XSLT (eXtensible Stylesheet Language Transform)

EJEMPLO:

```
<xsl:template match="//nombre">
```

```
  <html>
```

```
    <body>
```

```
      <h2> <xsl:value-of select="." /> </h2>
```

```
    </body>
```

```
  </html>
```

```
</xsl:template>
```

árbol de origen al que se aplica esta plantilla

salida

Entre les etiquetes d'inici i fi de la plantilla `xsl:template` s'escriu la transformació que hem de realitzar, és a dir, quin text i quines marques s'escriuran al document final de la transformació cada vegada que es trobi una instància amb l'element `//nombre` en el document origen.

Amb `<xsl:value-of ...>`, es recupera i escriu al document resultat el contingut de l'element que està sent processat.

# Estructura d'un document XSLT (eXtensible Stylesheet Language Transform)

## Document XML

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<?xml-stylesheet type="text/xsl" href="prueba.xsl" ?>
<ciudades>
  <ciudad>
    <nombre>Madrid</nombre>
    <habitantes>3500000</habitantes>
  </ciudad>
  <ciudad>
    <nombre>Huelva</nombre>
    <habitantes>150000</habitantes>
  </ciudad>
  <ciudad>
    <nombre>Toledo</nombre>
    <habitantes>50000</habitantes>
  </ciudad>
</ciudades>
```

Documento XML

Referencia al documento XSL

EL resultat:

```
<html>
  <head>
    <title>Ejemplo XSLT</title>
  </head>
  <body>
    <h1> CIUDADES DE ESPAÑA </h1>
    <h3>Madrid</h3>
    <h3>Málaga</h3>
    <h3>Toledo</h3>
  </body>
</html>
```

## Document XSL

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <head>
        <title>Ejemplo XSLT</title>
      </head>
      <body>
        <h1> CIUDADES DE ESPAÑA </h1>
        <xsl:apply-templates select="//nombre" />
      </body>
    </html>
  </xsl:template>

  <xsl:template match="nombre">
    <h3> <xsl:value-of select="." /> </h3>
  </xsl:template>
</xsl:stylesheet>
```

Documento prueba.xsl

Queremos obtener los nombres de las ciudades



# Estructura d'un document XSLT (eXtensible Stylesheet Language Transform)

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="template.xslt"?>
<catalog>
  <cd>
    <title>Private Dancer</title>
    <artist>Tina Turner</artist>
    <country>UK</country>
    <company>Capitol</company>
    <price>8.90</price>
    <year>1983</year>
  </cd>
  <cd>
    <title>Pavarotti Gala Concert</title>
    <artist>Luciano Pavarotti</artist>
    <country>UK</country>
    <company>DECCA</company>
    <price>9.90</price>
    <year>1991</year>
  </cd>
  <cd>
    <title>Unchain my heart</title>
    <artist>Joe Cocker</artist>
    <country>USA</country>
    <company>EMI</company>
    <price>8.20</price>
    <year>1987</year>
  </cd>
</catalog>
```

Document  
catalog.XML

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
    <body>
      <h2>My CD Collection</h2>
      <table border="1">
        <tr bgcolor="#9acd32">
          <th style="text-align:left">Title</th>
          <th style="text-align:left">Artist</th>
        </tr>
        <xsl:for-each select="catalog/cd">
          <tr>
            <td><xsl:value-of select="title"/></td>
            <td><xsl:value-of select="artist"/></td>
          </tr>
        </xsl:for-each>
      </table>
    </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

Document  
template.XSLT

El resultat:

## My CD Collection

Title	Artist
Private Dancer	Tina Turner
Pavarotti Gala Concert	Luciano Pavarotti
Unchain my heart	Joe Cocker

## Estructura d'un document XSLT

Un document XSLT s'escriu en XML i conté les regles de transformació per a convertir un document XML d'origen en un format desitjat. L'estructura bàsica d'un document XSLT inclou els següents components principals:

- **Declaració XML:** Igual que amb qualsevol document XML, un document XSLT comença amb la declaració XML estàndard. Per exemple: `<?xml version="1.0" encoding="UTF-8"?>`.
- **Element arrel `<xsl:stylesheet>` o `<xsl:transform>`:** Defineix el document com un full d'estil XSLT. Tots dos noms d'elements són intercanviables i han d'incloure l'atribut `version` per a especificar la versió de XSLT.
- **Elements `<xsl:template>`:** Són el cor del full d'estil XSLT, on es defineixen les regles de transformació. Cada `<xsl:template>` té un atribut `match` que utilitza XPath per a seleccionar els nodes del document XML d'origen que el template ha de processar.
- **Elements de sortida i transformació:** Dins dels templates, s'utilitzen diversos elements XSLT per a crear el contingut del document de sortida i manipular les dades del document d'origen. Alguns d'aquests elements inclouen `<xsl:value-of>`, `<xsl:for-each>`, i `<xsl:if>`.

## Elements i Atributs en XSLT

Els elements en XSLT defineixen com s'han de transformar les dades del document XML d'origen. Alguns dels elements més utilitzats són:

- **<xsl:value-of>**: Extreu el valor d'un node seleccionat i l'agrega al flux de sortida.
- **<xsl:for-each>**: Itera sobre una selecció de nodes i aplica el contingut de l'element **<xsl:for-each>** a cada node.
- **<xsl:if>** i **<xsl:choose>**: Permeten l'execució condicional basada en l'avaluació d'expressions XPath.

Els atributs en XSLT sovint especifiquen expressions XPath per a la selecció de nodes o defineixen característiques de la transformació, com en **match**, **test**, i **select**.

## XPath: Selecció de Nodes

XPath és un llenguatge que permet seleccionar nodes en documents XML. És utilitzat intensivament en XSLT per a identificar els nodes del document d'origen que s'han de transformar o sobre els quals s'ha d'actuar. Alguns conceptes clau de XPath inclouen:

- **Expressions de ruta:** Permeten navegar a través de l'estructura jeràrquica d'un document XML.

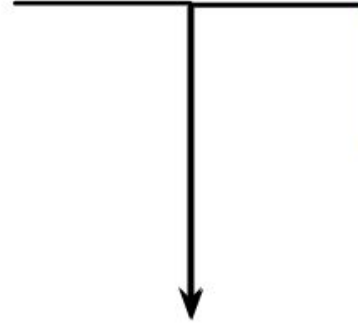
Per exemple, `//libre/autor` selecciona tots els nodes `<autor>` que són fills directes de `<libro>`.

- **Predicats:** S'utilitzen per a filtrar nodes basant-se en condicions.

Per exemple, `/llibre/autor[1]` selecciona el primer node `<autor>` sota cada node `<libro>`.

- **Funcions:** XPath inclou funcions per a cadenes, números, seqüències i més, que poden usar-se per a refinar seleccions o realitzar càlculs.

Comprendre com estructurar documents XSLT, com funcionen els seus elements i atributs, i com utilitzar XPath per a seleccionar i manipular nodes, és fonamental per a crear transformacions efectives i eficients amb XSLT.



**HTML**



Exercici 2: Aquest és un fitxer XML que conté informació sobre receptes:

1. Obriu el VSCode.
2. Crea una carpeta per guardar el teu nou projecte.
3. Crea dos arxius: receptes.xml i receptes.xsl
4. Executa Live Server i observa el que passa

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl"
href="receptes.xsl"?>
<receptes>
  <recepta>
    <titol>Hummus</titol>
    <temps>10 minuts</temps>
    <ingredients>
      <ingredient>Cigrons</ingredient>
      <ingredient>Oli d'oliva</ingredient>
      <ingredient>Llimona</ingredient>
    </ingredients>
  </recepta>
</receptes>
```

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Trans
form">
  <xsl:template match="/">
    <html>
      <head>
        <title>Receptes</title>
      </head>
      <body>
        <h1>Llista de receptes</h1>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html" encoding="UTF-8" indent="yes"/>
  <!-- Plantilla principal -->
  <xsl:template match="/">
    <html>
      <head>
        <title>Delicias Kitchen </ title>
      </ head>
      <body>
        <h1>Delicias Kitchen </ h1>
        <table border="1">
          <tr>
            <th>Imatge</th>
            <th>Títol</th>
            <th>Temps</th>
            <th>Ingredients</th>
          </ tr>
          < xsl:for-each select="receptes/recepta" >
            < tr>
              < td></td>
              < td><xsl:value-of select="títol" /></td>
              < td><xsl:value-of select="temps" /></td>
              < td><xsl:value-of select="ingredients" /></td>
            </ tr>
          </ xsl:for-each>
        </ table>
      </ body>
    </ html>
  </ xsl:template>
</ xsl:stylesheet>

```

Ara, proveu aquest altre XSL i observeu els resultats



## Com funciona?

Això que fem són un **TEMPLATE**, una plantilla.  
Així, per defecte, els fitxers de XSL s'acostumen a nomenar templates.xml

**match="/"**: Aquesta plantilla s'aplica a l'arrel del document XML.

Etiquetes HTML: El contingut es genera en format HTML (capçalera **<head>** i cos **<body>**).

## Segon pas: afegir el contingut dinàmic

Afegim el codi dins del **<body>**:

Per mostrar les dades del fitxer XML (per exemple, els títols de les receptes), utilitzem l'expressió **xsl:value-of**.

## Què fa aquest codi?

**<xsl:for-each>**: Recorre cada element **<recepta>** del document XML. **<xsl:value-of>**: Mostra el valor del camp seleccionat (per exemple, **<titol>** o **<temps>**). Resultat esperat en HTML: :

```
<xsl:template match="/">
  <html>
    <body>
      <h1>Llista de receptes</h1>
      <xsl:for-each select="receptes/recepta">
        <h2><xsl:value-of select="titol"/></h2>
        <p>Temps de preparació: <xsl:value-of
          select="temps"/></p>
      </xsl:for-each>
    </body>
  </html>
</xsl:template>
```

## Llistes i iteracions avançades


### Com transformar llistes d'ingredients?

Si volem mostrar els ingredients de cada recepta com una llista, utilitzem un altre `<xsl:for each>` per iterar dins dels ingredients.

```
<xsl:template match="/">
  <html>
    <body>
      <h1>Llista de receptes</h1>
      <xsl:for-each select="receptes/recepta">
        <h2><xsl:value-of select="titol"/></h2>
        <p>Tems de preparació: <xsl:value-of
          select="temps"/></p>
        <ul>
          <xsl:for-each select="ingredients/ingredient">
            <li><xsl:value-of select="."/></li>
          </xsl:for-each>
        </ul>
      </xsl:for-each>
    </body>
  </html>
</xsl:template>
```

# Ara, és el teu torn

## Delicias Kitchen

Imatge	Títol	Temps	Ingredients
	Hummus	10 minuts	Cigrons, Oli d'oliva, Llimona

Prova el codi anterior, descarrega una imatge i observa que està passant en la cel·la dels ingredients, apareix la ruta de la imatge? per què?

Ara amplia el xml amb dues receptes més i transforma-les.

Prova a posar una imatge a cadascuna de les receptes. Per fer-ho, necessitaràs crear una carpeta per guardar les imatges a dins.