***Puedes comprobar tus archivos en estas web***

XML

Colocar un XML

DTD

XSL

XPath

XQuery

JSON

RSS

Atom

También puedes usar este programa

Validar con Copy Editor

# XPath

**¿Qué es XPath?**

XPath es un lenguaje que se utiliza para navegar entre las etiquetas de un XML, se le pueden poner una serie de condiciones

**Ejemplo de XPath**

Para usar XPath necesitaremos un XML, donde pondremos la ruta de las etiquetas que queremos hacerlo

```
/videojuegos/videojuego
```

**Condicionales**

Para poder hacer una condición con XPath haremos lo siguiente

```
/ruta/de-la/etiqueta[condición]
```

- Atributos /ruta/etiqueta/@atributo
- Posición /ruta/etiqueta[num_posición]
- Última posición /ruta/etiqueta[last()]
- Igualar atributo /ruta/etiqueta[@atributo="igualar"]
- Solo texto /ruta/etiqueta/text()
- Operadores /ruta/etiqueta[@atributo>numero]
- Dos condiciones /ruta/etiqueta[@atributo=>numero and @atributo<=numero]

- Contener /ruta/etiqueta[contains(etiqueta_contener, “que_debe_contener”)]/etiqueta

**¿Te ha explotado la cabeza?**

No te preocupes, aquí te dejo unos ejemplos para que no sufras tanto

```
/videojuegos/videojuego[@anyo>=2000]/titulo
/videojuegos/videojuego[2]
/videojuegos/videojuego[contains(titulo, 'G')]/titulo
/videojuegos/videojuego/@PEGI
```

Ver otros condicionalesonales para que solo se muestren datos que coincidan con las condiciones que pongamos. Las condiciones se declaran en la parte superior. Aquí tienes un ejemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="inventario">
    <html>
        <head></head>
        <body>
            <xsl:apply-templates select="producto[lugar/@edificio='A' and lugar/
            <xsl:apply-templates select="producto[peso>'7']"/>
            <xsl:apply-templates select="producto[lugar/@edificio='A' and peso &
            <xsl:sort select="autores/autor" order="descending"/>
        </body>
    </html>
</xsl:template>
<xsl:template match="producto">
    <ul>
        <li><xsl:value-of select="peso"/></li>
        <li><xsl:value-of select="nombre"/> <xsl:value-of select="lugar"/></li>
    </ul>
</xsl:template>
</xsl:stylesheet>
```

Hay algunos condicionales que necesitan que estén “encapsulados” en la etiqueta_sobre_la_que_trabajamos. Aquí tienes un ejemplo
Como vemos para usar el orden descendente se necesita encerrar a la condición

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="catalogo">
    <html>
```

```
        <head></head>
        <body>
            <xsl:apply-templates select="libro">
            <xsl:sort select="autores/autor" order="descending"/>
        </xsl:apply-templates>
        </body>
    </html>
</xsl:template>
<xsl:template match="libro">
    <ol>
        <li>Autor: <xsl:value-of select="autores/autor"/></li>
    </ol>
</xsl:template>
</xsl:stylesheet>
```

También podemos crear otros tipo de archivo mediante un XSL, como un 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="listado">
            <xsl:apply-templates select="cuenta"/>
</xsl:template>
<xsl:template match="cuenta">
    <datos>
        <cuentas>
            <cuenta>
                <xsl:attribute name="dnititular">
                    <xsl:value-of select="titular/@dni"/>
                </xsl:attribute>
                <creacion><xsl:value-of select="fechacreacion"/></creacion>
                <titular><xsl:value-of select="titular"/></titular>
                <saldoactual><xsl:value-of  select="saldoactual"/> euros</saldoa
            </cuenta>
        </cuentas>
    </datos>
</xsl:template>
</xsl:stylesheet>
```