

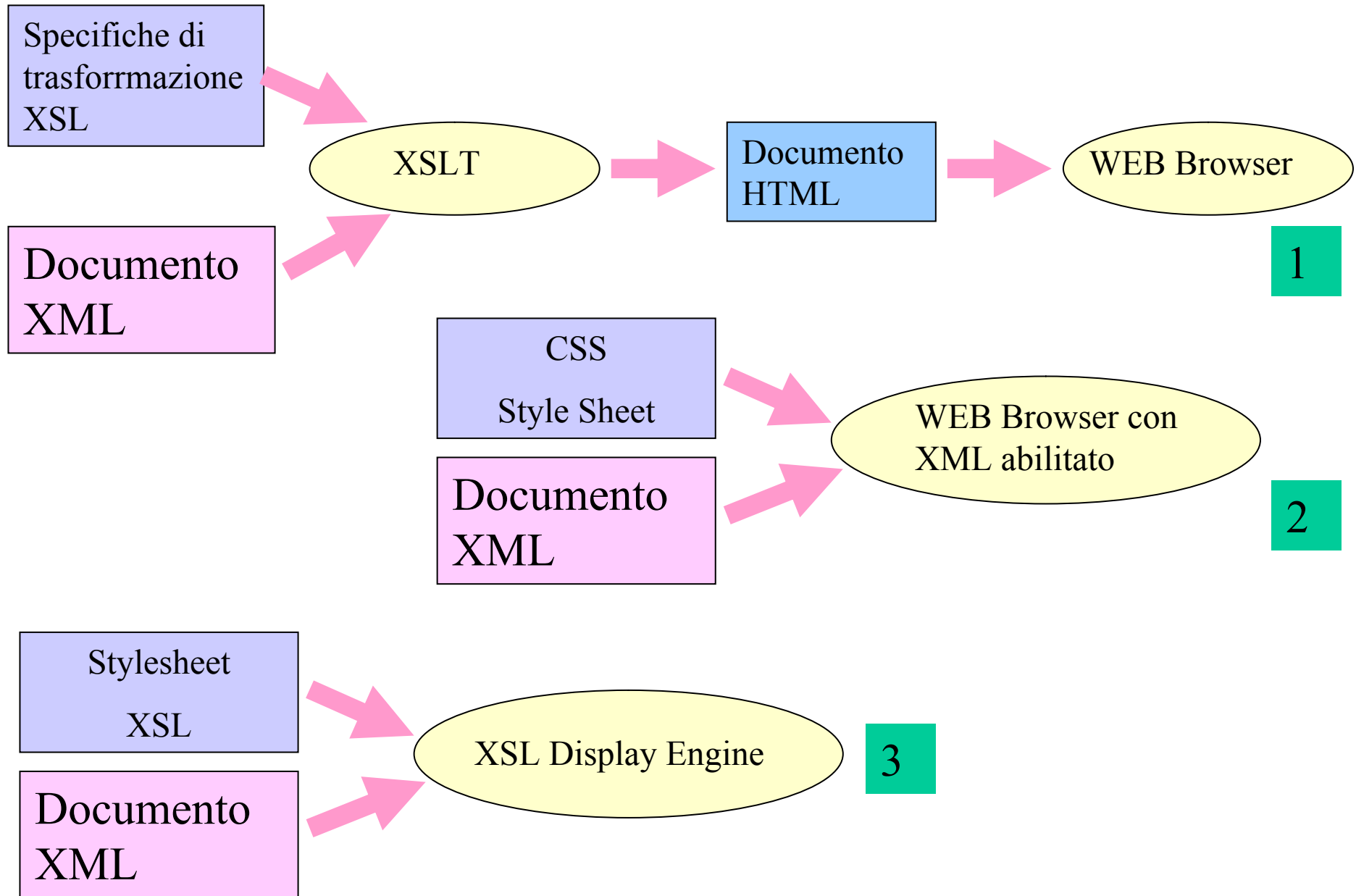
Introduzione al linguaggio

XSL (eXensible Styles Language)

XSLT(eXensible Styles Language Trasformations)

Lezione del Corso Interazione Uomo Macchina 1, Docente Francesco Mele  
Corso di Laurea in Informatica Università di Napoli Federico II,  
Anno Accademico 2003-2004

# Tre strade per visualizzare (interpretare) XML su un browser Web



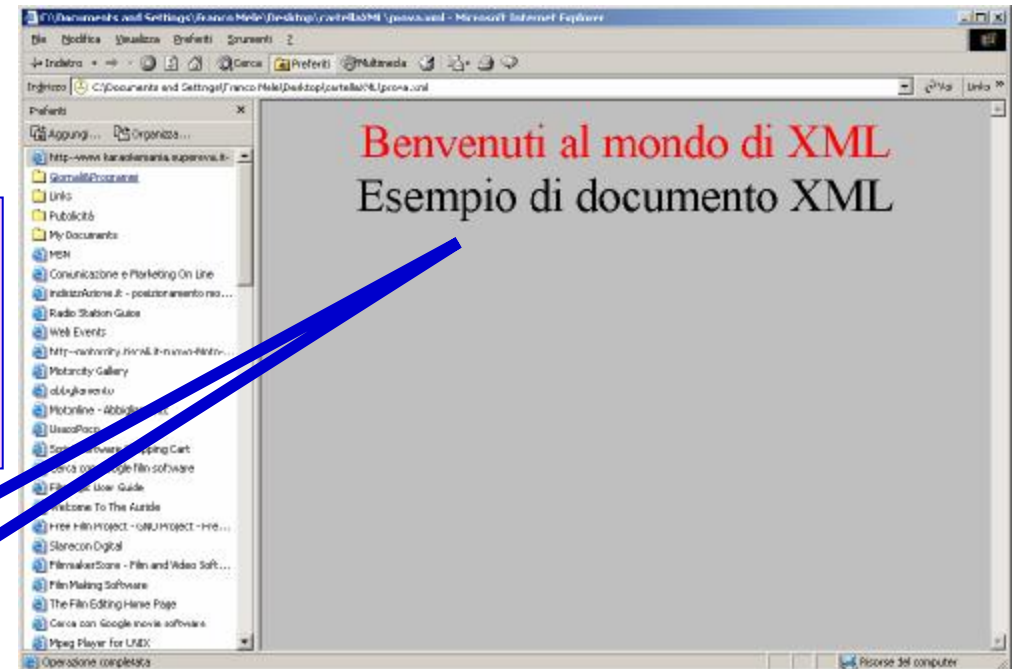
# Un esempio: possibilità di definire linguaggi di style sheet

```
<?xml version="1.0" standalone="yes"?>
<?xml-stylesheet type="text/css" href="greeting.css"?>

<DOCUMENT>
  <benvenuto> Benvenuti al mondo di XML</benvenuto>
  <messaggio> Esempio di documento XML </messaggio>
</DOCUMENT>
```

```
benvenuto {display: block; font-size: 36pt;
            color: #FF0000; text-align: center}
messaggio {display: block; font-size: 36pt;
           color: #000000; text-align: center}
```

Rendering



XSL (eXensible Styles Language)

XSLT(eXensible Styles Language Trasformation)

XSL è composto da due parti - quindi da due linguaggi pressoché indipendenti da un punto di vista funzionale:

- il linguaggio di **formattazione** che consente l'applicazione degli stili ai documenti;
- il linguaggio di **trasformazione** che permette di trasformare documenti tra forme diverse

I documenti XSLT sono documenti XML ben formati:

- cominciano con la dichiarazione iniziale

```
<?xml version=1.0">
```

- sono stati standardizzati: usano il namespace XSLT

```
'http://www.w3.org/1999/XSL/Transform'
```

## Esempio di inizio documento XSLT

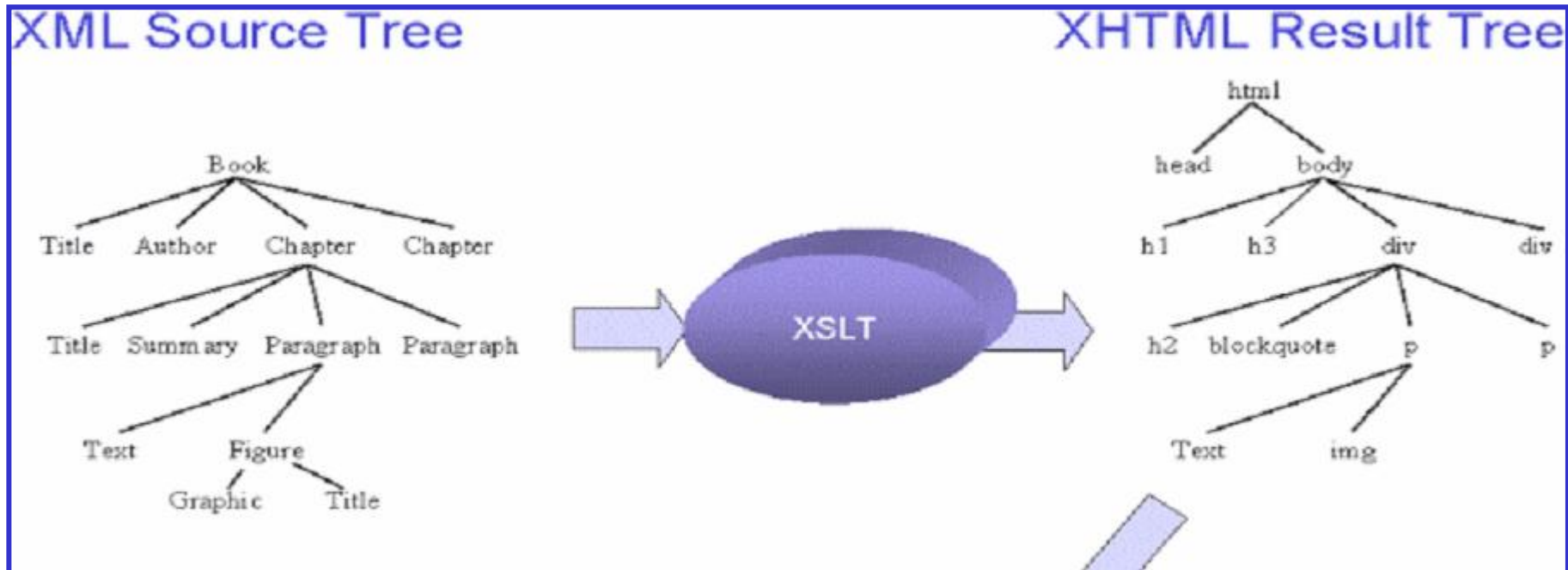
```
<?xml version=1.0">
```

```
<xsl:stylesheet version='1.0'
```

```
xmlns:xsl='http://www.w3.org/1999/XSL/Transform' >
```

- 
- 
-

Le trasformazioni XSLT accettano un albero come input e producono una struttura ad albero come output.



Esempi tipici di trasformazioni:

XML --> HTML

XML --> XML

## Come avvengono le trasformazioni?

... ricordando che un documento XML è costituito in genere da n nodi - XSLT usa dei modelli (templates)

Ogni template:

- 1- seleziona un nodo della struttura del documento XML;
- 2- indica al processore XSLT come trasformare il nodo selezionato in 1 (ossia indica quale sostituzione deve essere eseguita - introducendo il codice di un altro linguaggio o di quello di XML stesso).

```
esempio1 - Notepad
File Edit Format Help
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="trasformazione1.xsl"?>
  <Radice>
    <Elemento1>Contenuto1
  </Elemento1>
    <Elemento2>Contenuto2
  </Elemento2>
</Radice>
```

```
C:\Documents and Settings\Franco Mele\Desktop\cartellaXML\trasformazione1.
File Modifica Visualizza Preferiti Strumenti ?
Indietro
Indirizzo C:\Documents and Settings\Franco Mele\Desktop\cartellaXML\trasformazione1.
- <xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
- <xsl:template match="Radice">
- <HTML>
- <HEAD>
  <TITLE>Una banale trasformazione</TITLE>
</HEAD>
- <Body>
  <H1>Trasformazione banale effettuata</H1>
</Body>
</HTML>
</xsl:template>
</xsl:stylesheet>
```

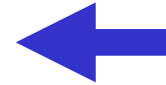
Una prima  
semplice  
trasformazione

## Programma XSL

```
trasformazione1 - Notepad
File Edit Format Help
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform'>
<xsl:template match="Radice">
  <HTML>
    <HEAD>
      <TITLE>
        Una banale trasformazione
      </TITLE>
    </HEAD>
    <Body>
      <H1>Trasformazione banale effettuata</H1>
    </Body>
  </HTML>
</xsl:template>
</xsl:stylesheet>
```

## Documento XML

```
esempio1 - Notepad
File Edit Format Help
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="trasformazione1.xsl"?>
  <Radice>
    <Elemento1>Contenuto1
  </Elemento1>
    <Elemento2>Contenuto2
  </Elemento2>
  </Radice>
```



<HTML>

Codice HTML

<HEAD>

<TITLE>

Una banale trasformazione

</TITLE>

</HEAD>

<Body>

<H1>

Trasformazione banale effettuata

</H1>

</Body>

</HTML>



## Rendering



## esempio1 - Notepad

File Edit Format Help

```
<?xml version="1.0"
<?xml-stylesheet type="text/xsl" href="trasformazione1.xsl" ?>
  <Radice>
    <Elemento1>Contenuto1
  </Elemento1>
    <Elemento2>Contenuto2
  </Elemento2>
</Radice>
```

Un template che match con radice:

**<xsl:template match="/">**

## trasformazione1 - Notepad

File Edit Format Help

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1
xmlns="http://www.w3.org/1
<xsl:template match="Radice">
  <HTML>
    <HEAD>
      <TITLE>
        Una banale tra
      </TITLE>
    </HEAD>
    <Body>
      <H1>Trasformazione ban
    </Body>
  </HTML>
</xsl:template>
</xsl:stylesheet>
```

## trasf\_radice - Notepad

File Edit Format Help

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/>
xmlns="http://www.w3.org/TR/xhtml1,
<xsl:template match="/">
  <HTML>
    <HEAD>
      <TITLE>
        Una banale trasforn
      </TITLE>
    </HEAD>
    <Body>|
      Trasformazione effettuata
    </Body>
  </HTML>
</xsl:template>
```

# Ottenere un valore attraverso xsl:value-of

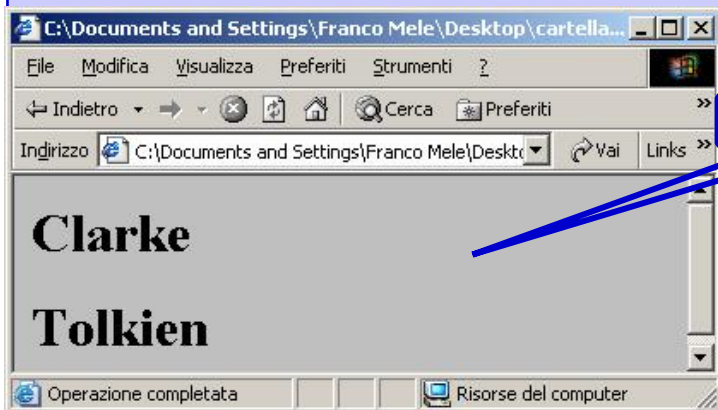
```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="trasformazione3.xsl"?>
  <libri>
    <libro>
      <titolo>2001: Odissea nello spazio</titolo>
      <autore>Clarke</autore>
      <editore>Rizzoli</editore>
    </libro>
    <libro>
      <titolo>Il signore degli anelli</titolo>
      <autore>Tolkien</autore>
      <editore>Rizzoli</editore>
    </libro>
  </libri>
```

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Trans

<xsl:template match="libri">
<HTML>
<xsl:apply-templates/>
</HTML>
</xsl:template>

<xsl:template match="libro">
<H1><xsl:value-of select="autore"/></H1>
</xsl:template>

</xsl:stylesheet>
```



rendering

# Gestire selezioni multiple con xsl:for-each

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="trasformaz
<libri>
  <libro>
    <titolo>La societ  delle menti</titolo>
    <autore>Castelfranchi</autore>
    <autore>Conte</autore>
    <editore>Utet Libreria</editore>
  </libro>
  . . . .
</libro>
</libri>
```

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Tran

<xsl:template match="libri">
<HTML>
<xsl:apply-templates/>
</HTML>
</xsl:template>

<xsl:for-each select="autore">
  <H1>
    <xsl:value-of select="."/>
  </H1>
</xsl:template>
</xsl:stylesheet>
```



# Pattern per l'attributo match per i figli

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="trasformazione3.xsl"?>
  <libri>
    <libro>
      <titolo>2001: Odissea nello spazio</titolo>
      <autore>Clarke</autore>
      <editore>Rizzoli</editore>
    </libro>
    <libro>
      <titolo>Il signore degli anelli</titolo>
      <autore>Tolkien</autore>
      <editore>Rizzoli</editore>
    </libro>
  </libri>
```

Gioca il ruolo di nodo corrente

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform' >

  <xsl:template match="libri/libro">
    <H1>
      <xsl:value-of select="."/>
    </H1>
  </xsl:template>

</xsl:stylesheet>
```

# Pattern per l'attributo match per discendenti

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="trasformazione3.xsl"?>
  <libri>
    <libro>
      <titolo>2001: Odissea nello spazio</titolo>
      <autore>Clarke</autore>
      <editore>Rizzoli</editore>
    </libro>
    <libro>
      <titolo>Il signore degli anelli</titolo>
      <autore>Tolkien</autore>
      <editore>Rizzoli</editore>
    </libro>
  </libri>
```

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform' >

<xsl:template match="libri//titolo">
  <H1>
    <xsl:value-of select="."/>
  </H1>
</xsl:template>

</xsl:stylesheet>
```

## Più pattern possibili: l'operatore OR " | "

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="trasformazione3.xsl"?>
  <libri>
    <libro>
      <titolo>2001: Odissea nello spazio</titolo>
      <autore>Clarke</autore>
      <editore>Rizzoli</editore>
    </libro>
    <libro>
      <titolo>Il signore degli anelli</titolo>
      <autore>Tolkien</autore>
      <editore>Rizzoli</editore>
    </libro>
  </libri>
```

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform' >
  <xsl:template match="titolo | autore">
    <H1>
      <xsl:apply-templates/>
    </H1>
  </xsl:template>
</xsl:stylesheet>
```

## Fare test con [ ]

L'operatore [ ] può essere usato per saggiare se una certa condizione è vera. Ad esempio si possono **effettuare** test su:

- il valore di un attributo in una data stringa;
- il valore di un elemento;
- se un elemento contiene un particolare figlio, attributo o altro elemento

### Esempio:

```
<xsl:template match = "libro[editore]">
```

Questa espressione fa corrispondere gli elementi <libro> che hanno un elemento figlio editore.

Specificare un pattern per l'attributo **select**

L'attributo **select** usa le espressioni **Xpath** –  
raccomandate da W3C ([www.w3.org/TR/xpath](http://www.w3.org/TR/xpath))

# La sintassi abbreviata di Xpath: l'esempio di "//"

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="stylelibri.xsl"?>
  <libro>
    <titolo>2001: Odissea nello spazio</titolo>
    <autore>Clarke</autore>
    <editore>Rizzoli</editore>
  </libro>
```

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform' >

<xsl:template match="/">
<H2><xsl:value-of select="//autore"/></H2>
<H1><xsl:value-of select="//titolo"/></H1>
</xsl:template>
</xsl:stylesheet>
```



Rendering

## Ordinare elementi: xsl:sort

```
<?xml version="1.0" encod  
<?xml-stylesheet type="tex
```

```
<libri>
```

```
<libro>
```

```
<titolo>2001: Odissea nello spazio</titolo>
```

```
<autore>Clarke</autore>
```

```
<editore>Rizzoli</editore>
```

```
</libro>
```

```
<libro>
```

```
<titolo>Il signore degli anelli</titolo>
```

```
<autore>Tolkien</autore>
```

```
<editore>Rizzoli</editore>
```

```
</libro>
```

```
</libri>
```

```
...
```

```
<xsl:apply-templates>
```

```
<H1>
```

```
<xsl:sort select="autore"/>
```

```
</H1>
```

```
</xsl:apply-templates>
```

Esiste un buon (dal punto di vista didattico) tutorial elettronico

**Zvon XSL Tutorial**

[http://www.zvon.org/xxl/XSLTutorial/Old\\_version/Books/Book1/index.html](http://www.zvon.org/xxl/XSLTutorial/Old_version/Books/Book1/index.html)

# Zvon XSL Tutorial [10]

**Zvon XSL Tutorial - Netscape**  
File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Shop Stop

Bookmarks Location: <http://zvon.org/xml/XSLTutorial/Books/Book1/index.html> What's Related

**I** **#** **S** **K**

**Keywords**

\*: 8  
/: 1 4 73 74  
//: 1 4 59  
@: 2 12 13 59  
@case-order: 15  
@data-type: 14  
@format: 18 21  
@level: 19 21  
@mode: 26  
Infinity: 44  
XPath: 72  
ancestor: 27 58  
ancestor-or-self: 27 58  
attribute: 27 58 59  
boolean(): 23 25 40 41  
ceiling(): 38  
child: 27 58 59

**List of examples**

1 2 3 4 5 6 7 8 9 10 11 12 13 14  
15 16 17 18 19 20 21 22 23 24  
25 26 27 28 29 30 31 32 33 34  
35 36 37 38 39 40 41 42 43 44  
45 46 47 48 49 50 51 52 53 54

**About...**

This program was written as a part of **Zvon project**.  
Author: **Miloslav Nic**  
There is now a complete **XSLT reference and interactive V**

**HELP AND HOW TO CONTRIBUTE**

Click on the title to open Help in a new window. This is a general behaviour of all windows. **You can download entire tutorial**

**K** **A** **Z**

**Nic Miloslav**

**Introduction to XSL \*\*\***

**Introduction** sum  
**Templates** sum  
**Attributes** sum  
**Axes** sum  
**Repetition and sorting** sum  
**Creation of elements and attributes** sum  
**Conditional processing** sum  
**Numbers generation and formatting** sum  
**Variables** sum  
**Numeric calculations** sum  
**Boolean functions** sum  
**String functions** sum  
**Node set functions** sum  
**Output** sum

Document: Done

**XML source**

```
<?xml version="1.0"?>
<xslTutorial >
<title>XSL</title>
<author>John Smith</author>
</xslTutorial>
```

**HTML output 1**

```
<HTML>
<HEAD> </HEAD>
<BODY>
<H1>XSL</H1>
<H2>John Smith</H2> </BODY>
</HTML>
```

**XSL****John Smith****XSL stylesheet 1**

```
<xsl:stylesheet version='1.0'
xmlns:xsl='http://www.w3.org/1999/XSL/Transform' >
<xsl:template match="/">
<H1><xsl:value-of select="//title"/></H1>
<H2><xsl:value-of select="//author"/></H2>
</xsl:template>
</xsl:stylesheet>
```



Rendering