Laboratorio di Basi di Dati e Multimedia

Laurea in Informatica Multimediale

Docente: Carlo Combi

Email: carlo.combi@univr.it

Lezione 8

Il linguaggio XML

eXtensible Markup Language

- XML è un linguaggio di marcatura proposto dal W3C
- XML definisce una sintassi generica per contrassegnare i dati di un documento elettronico con marcatori (tag) semplici e leggibili
- La sintassi XML viene utilizzata in contesti molto diversi:
  - ✓ pagine web
  - ✓ scambio di dati elettronici
  - ✓ grafica vettoriale
  - ✓ cataloghi di prodotti
  - ✓ sistemi di gestione di messaggi vocali
  - ✓ …
- http://www.w3c.org/XML

XML: evoluzione

- 1986: Standard Generalized Markup Language (SGML)
  - ✓ Linguaggio di marcatura strutturato, per la rappresentazione elettronica di documenti di tipo testuale
- 1995: HyperText Markup Language (HTML)
  - ✓ Applicazione di SGML che permette di descrivere come il contenuto di un documento verrà presentato
- 1998: eXtensible Markup Language (XML)
  - ✓ Versione “leggera” di SGML, che consente una formattazione semplice e molto flessibile.

HTML vs XML

- HTML
  - Insieme fisso di tag
  - Descrizione di come il documento verrà presentato
  - Usato solo per la costruzione di pagine web

- XML
  - Insieme non fisso di tag: i tag possono essere personalizzati
  - Descrizione del contenuto del documento
  - Usato in molti domini diversi

HTML vs XML: esempio

```
<h1>
    Essential XML: <br>
        Oltre il Markup
</h1>

<ul>
    <li>
        Don Box
    </li>
    <li>
        Aaron Skonnard
    </li>
    <li>
        John Lam
    </li>

</ul>

<i>
    Addison-Wesley
</i>
```

```
<titolo>
    Essential XML:
        Oltre il Markup
</titolo>

<autore>
    Don Box
</autore>
<autore>
    Aaron Skonnard
</autore>
<autore>
    John Lam
</autore>
<casa editrice>
    Addison-Wesley
</casa editrice>
```

Documenti *ben formati*

- XML è più restrittivo di HTML per quanto riguarda il posizionamento dei tag e il modo in cui vengono scritti

- Ogni documento XML deve essere *ben formato*
  - ✓ Ha una sola radice
  - ✓ L'annidamento dei tag deve essere corretto
  - ✓ Tutti i tag aperti devono essere chiusi
  - ✓ I valori degli attributi devono essere specificati tra virgolette

Esempio di documento XML: persona.xml

- Un documento XML è un file di testo

Sintassi dei tag

- Tag iniziale

  <nome>
- Tag finale

  </nome>
- Elementi vuoti: sono elementi privi di contenuto

  <maschio></maschio> oppure <maschio/>

- XML è sensibile alla differenza tra maiuscole e minuscole (è *case-sensitive*)

Contenuto di un elemento

- Contenuto di tipo carattere

```
<nome> Luca Rossi </nome>
```

- Contenuto costituito da altri elementi (figli)

```
<indirizzo>
        <via> Via Mazzini </via>
        <civico> 10 </civico>
        <città> Verona </città>
</indirizzo>
```

Contenuto di un elemento (2)

- Contenuto misto

```
<dati_anagrafici> Il signor
  <persona>
        <nome> Mario Rossi </nome> vive in
        <indirizzo>
              <via> Via Mazzini </via>
              <civico> 10 </civico>
              <città> Verona </città>
        </indirizzo>
  </persona>
</dati_anagrafici>
```

Attributi

- Un attributo consiste in una coppia nome-valore associata al tag iniziale di un elemento

```
<persona cod_fisc="RSSMRA65E25L781T">
```

- Si possono usare anche gli apici singoli

```
<persona cod_fisc='RSSMRA65E25L781T'>
```

Nomi XML

- Possono essere costituiti da qualsiasi carattere alfanumerico
- Possono includere:
  - ✓ Underscore _
  - ✓ Trattino -
  - ✓ Punto .
- Possono iniziare solo con lettere, ideogrammi o con il carattere underscore
- Non possono includere:
  - ✓ Altri caratteri di punteggiatura
  - ✓ Virgolette
  - ✓ Apostrofi
  - ✓ $ e %
  - ✓ < e >
  - ✓ Spazi

Esempi di nomi XML

- Nomi ben formati:
  - ✓ <Nome_persona> Maria </Nome_persona>
  - ✓ <Giorno-Mese-Anno> 10/06/2004 </Giorno-Mese-Anno>
  - ✓ <_indirizzo> Via Stella 10 </_indirizzo>

- Nomi NON ben formati:
  - ✓ <Nome persona> Maria </Nome persona>
  - ✓ <Giorno/Mese/Anno> 10/06/2004 </Giorno/Mese/Anno>
  - ✓ <citta'> Verona </citta'>
  - ✓ <1_telefono> 045 1234567 </1_telefono>
  - ✓ <%vendita> 20 </%vendita>

La dichiarazione XML

- I documenti XML dovrebbero iniziare con una dichiarazione XML

```
<?xml version="1.0" encoding="US-ASCII" standalone="yes"?>
```

Document Type Definition (DTD)

- Un DTD descrive la struttura di un documento XML:
  - ✓i tag ammessi
  - ✓le regole di annidamento dei tag

- I DTD vengono utilizzati per la validazione di un documento XML
  - ✓Un documento XML è valido quando è conforme ad un dato DTD

Esempio di DTD: elenco.dtd

```
<!ELEMENT elenco (libro+)>
<!ELEMENT libro (titolo,prezzo?)>
<!ELEMENT titolo (#PCDATA)>
<!ELEMENT prezzo (#PCDATA)>
```

- ✓ Ogni riga rappresenta la dichiarazione di elemento
- ✓ L'elemento elenco contiene uno o più elementi libro
- ✓ L'elemento libro contiene un titolo e zero o un prezzo
- ✓ titolo e prezzo possono contenere solo testo

Dichiarazione di elementi

<!ELEMENT nome_elemento (modello_di_contenuto)>

- ✓ Il modello di contenuto può essere:
  - #PCDATA
  - Elementi figli
  - Sequenze
  - Misto
  - EMPTY
  - ANY

Modello di contenuto: #PCDATA

- Specifica che l'elemento deve contenere solamente dati di tipo carattere

- L'elemento non può contenere elementi figli di alcun tipo

Esempio: L'elemento titolo deve contenere solo testo

<!ELEMENT titolo (#PCDATA)>

Modello di contenuto: Elementi Figli

- Specifica che un elemento deve contenere esattamente un elemento figlio di un determinato tipo

Esempio: L'elemento libro deve contenere esattamente un elemento titolo (né più né meno di uno)

<!ELEMENT libro (titolo)>

Modello di contenuto: Sequenze

- Specifica che un elemento deve contenere più elementi figli
- I figli vengono elencati in una sequenza separati da virgole
- Gli elementi figli devono apparire all'interno dell'elemento padre nell'ordine specificato

Esempio: L'elemento libro deve contenere un elemento titolo e un elemento prezzo

<!ELEMENT libro (titolo,prezzo)>

Il numero di figli

- Per indicare quante istanze di un elemento possono apparire si usa:
  - ✓ ? zero o una istanza
  - ✓ * zero o più istanze
  - ✓ + una o più istanze

Esempio: L'elemento libro deve contenere un elemento titolo, uno o più elementi autore e zero o un elemento prezzo

<!ELEMENT libro (titolo,autore+,prezzo?)>

Scelte

- Una scelta è un elenco di nomi di elementi (due o più) che possono apparire nell'elemento padre
- Gli elementi della scelta vengono separati da barre verticali
- L'elemento padre non può contenere entrambi gli elementi elencati nella scelta

Esempio: L'elemento contatto può contenere o un elemento telefono_casa o un elemento telefono_ufficio

```
<!ELEMENT contatto (telefono_casa |
telefono_ufficio)>
```

Parentesi

- Per combinare scelte e sequenze si possono usare le parentesi

Esempio: L'elemento indirizzo deve contenere un elemento tra via e piazza e un elemento civico

<!ELEMENT indirizzo ((via | piazza), civico)>

L'elemento persona può contenere un elemento nome e un elemento cognome o un elemento cognome e un elemento nome

<!ELEMENT persona ((nome,cognome) | (cognome,nome))>

Modello di contenuto: Misto

- Specifica che un elemento deve contenere sia dati di tipo carattere che elementi figli
- Non è possibile specificare:
  - ✓l'ordine in cui appariranno
  - ✓quante istanze di essi appariranno
- L'elemento #PCDATA deve essere il primo della lista

Esempio: L'elemento libro può contenere dati di tipo carattere e elementi figli titolo e prezzo

```
<!ELEMENT libro (#PCDATA|titolo|prezzo)*>
```

Modello di contenuto: EMPTY

- Specifica che un elemento deve essere vuoto e quindi senza nessun tipo di contenuto

Esempio: L'elemento immagine deve essere un elemento vuoto

<!ELEMENT immagine EMPTY>

Modello di contenuto: ANY

- Specifica che un elemento può contenere qualsiasi cosa
  - ✓ Testo
  - ✓ Elementi figli
  - ✓ Contenuto misto
- Gli elementi che appaiono come figli devono comunque esser stati dichiarati

Esempio: L'elemento pagina può contenere qualsiasi cosa

<!ELEMENT pagina ANY>

Dichiarazione di attributi

```
<!ATTLIST nome_elemento
          nome_attributo1 CDATA #REQUIRED
          nome_attributo2 CDATA #IMPLIED
          nome_attributo3 CDATA #FIXED valore>
```

Esempio: L'elemento immagine ha un attributo codice obbligatorio ed un attributo titolo opzionale

```
<!ATTLIST immagine codice CDATA #REQUIRED
                  titolo CDATA #IMPLIED>
```

Valori di default per gli attributi

- #IMPLIED: il valore dell'attributo è opzionale
- #REQUIRED: il valore dell'attributo è obbligatorio
- #FIXED: il valore dell'attributo è costante e immutabile
- Literal: indica il valore di default sotto forma di stringa tra apici

Tipi di Attributi

- ✓ CDATA
- ✓ NMTOKEN
- ✓ NMTOKENS
- ✓ Enumerazione
- ✓ ID
- ✓ IDREF
- ✓ IDREFS
- ✓ ENTITY
- ✓ ENTITIES
- ✓ NOTATION

Tipi di Attributi (1)

- CDATA: può contenere qualsiasi tipo di stringa accettabile in un documento XML ben formato

```
<!ATTLIST immagine titolo CDATA #IMPLIED>

<immagine titolo="tramonto"/>
```

- NMTOKEN: può iniziare con qualsiasi carattere

```
<!ATTLIST libro anno_pubblicazione NMTOKEN #REQUIRED>

<libro anno_pubblicazione="1950 d.c."/>
```

- NMTOKENS: può contenere uno o più token

```
<!ATTLIST esibizione date NMTOKENS #IMPLIED>

<esibizione date="10-07-2004 17-07-2004 24-07-2004"/>
```

Tipi di Attributi (2)

- Enumerazione: lista di tutti i possibili valori assegnabili all'attributo

```
<!ATTLIST data giorno (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) #REQUIRED
           mese (gennaio|febbraio|marzo|aprile|maggio|
                 giugno|luglio|agosto|settembre|ottobre|
                 novembre|dicembre) #REQUIRED
           anno (2003|2004|2005|2006|2007) #REQUIRED>
```

```
<data giorno="15" mese="agosto" anno="2007"/>
```

Tipi di Attributi (3)

- ID: contiene un nome XML che ha valore univoco all'interno del documento

```
<!ATTLIST persona cod_fisc ID #REQUIRED>

<persona cod_fisc="RSSMRA65E25L781T"/>
```

- IDREF: riferimento all'attributo di tipo ID di un elemento del documento

```
<!ATTLIST persona cod_fisc ID #REQUIRED>
<!ATTLIST docente persona IDREF #REQUIRED>

<persona cod_fisc="RSSMRA65E25L781T"/>
<docente persona="RSSMRA65E25L781T"/>
```

Tipi di Attributi (4)

- IDREFS: contiene una lista di nomi XML ognuno dei quali deve essere un ID valido di un elemento del documento

```
<!ATTLIST persona cod_fisc ID #REQUIRED>
<!ATTLIST sposi persone IDREFS #REQUIRED>

<persona cod_fisc="RSSMRA65E25L781T"/>
<persona cod_fisc="BNCFRN63D45L781T"/>
<sposi persone="RSSMRA65E25L781T
   BNCFRN63D45L781T"/>
```

Validazione

- Un documento XML per il quale è richiesta la validazione deve includere un riferimento al DTD con cui deve essere messo a confronto
- Il riferimento deve essere fornito nella dichiarazione del tipo di documento

  <!DOCTYPE elenco SYSTEM "http://ibiblio.org/xml/dtds/elenco.dtd">
- Questa dichiarazione afferma che elenco è la radice del documento e il DTD si trova all'URL http://ibiblio.org/xml/dtds/elenco.dtd
- La dichiarazione del tipo di documento si trova dopo la dichiarazione XML

Dichiarazione del tipo di documento

- Se il DTD di riferimento si trova ad un certo URL:

  <!DOCTYPE elenco SYSTEM "http://ibiblio.org/xml/dtds/elenco.dtd">

- Se il DTD si trova allo stesso URL del documento:

  <!DOCTYPE elenco SYSTEM "/dtds/elenco.dtd">

- Se il DTD si trova nella stessa directory del documento

  <!DOCTYPE elenco SYSTEM "elenco.dtd">

Esempio di documento valido

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE elenco SYSTEM "elenco.dtd">
<elenco>
   <libro>
        <titolo> XML Guida di riferimento </titolo>
        <prezzo> 35 </prezzo>
   </libro>

   <libro>
        <titolo> Basi di dati </titolo>
   </libro>
</elenco>
```

Dichiarazione del tipo di documento (2)

- Il DTD può essere inserito direttamente nella dichiarazione del tipo di documento

```
<?xml version="1.0"?>
<!DOCTYPE elenco [
    <!ELEMENT elenco (libro+)>
    <!ELEMENT libro (titolo,prezzo?)>
    <!ELEMENT titolo (#PCDATA)>
    <!ELEMENT prezzo (#PCDATA)>
]>
<elenco>
    <libro><titolo> XML Guida di riferimento </titolo>
            <prezzo> 35 </prezzo>
    </libro>
    <libro><titolo> Basi di dati </titolo>
    </libro>
</elenco>
```

Validazione di un documento

- Validatori online:
  - Servizio di validazione del W3C

    http://validator.w3.org/
  - Verificatore di ben formazione XML e validatore di Richard Tobin

    http://www.cogsci.ed.ac.uk/%7Erichard/xml-check.html
- Per utilizzare questi validatori i documenti e i DTD associati possono essere su un server Web accessibile al pubblico o inviati all'applicazione in vari modi.

Esempio: file prova.xml

http://validator.w3.org/

```
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<!DOCTYPE elenco
[<!ELEMENT elenco (libro+)>
<!ELEMENT libro (titolo,prezzo?)>
<!ELEMENT titolo (#PCDATA)>
<!ELEMENT prezzo (#PCDATA)>
]>
<elenco>
        <libro>
                <titolo> XML Guida di riferimento </titolo>
                <prezzo> 35 </prezzo>
        </libro>
        <libro>
                <titolo> Basi di dati </titolo>
        </libro>
</elenco>
```

Esempio: file libri1.xml ed elenco.dtd

http://validator.w3.org/

```
<?xml version='1.0' encoding='UTF-8' standalone='no'?>
<!DOCTYPE elenco SYSTEM 'elenco.dtd'>
<elenco>
        <libro>
                <titolo> XML Guida di riferimento </titolo>
                <prezzo> 35 </prezzo>
        </libro>
        <libro>
                <titolo> Basi di dati </titolo>
        </libro>
</elenco>
```

libri1.xml

```
<!ELEMENT elenco (libro+)>
<!ELEMENT libro (titolo,prezzo?)>
<!ELEMENT titolo (#PCDATA)>
<!ELEMENT prezzo (#PCDATA)>
```

elenco.dtd

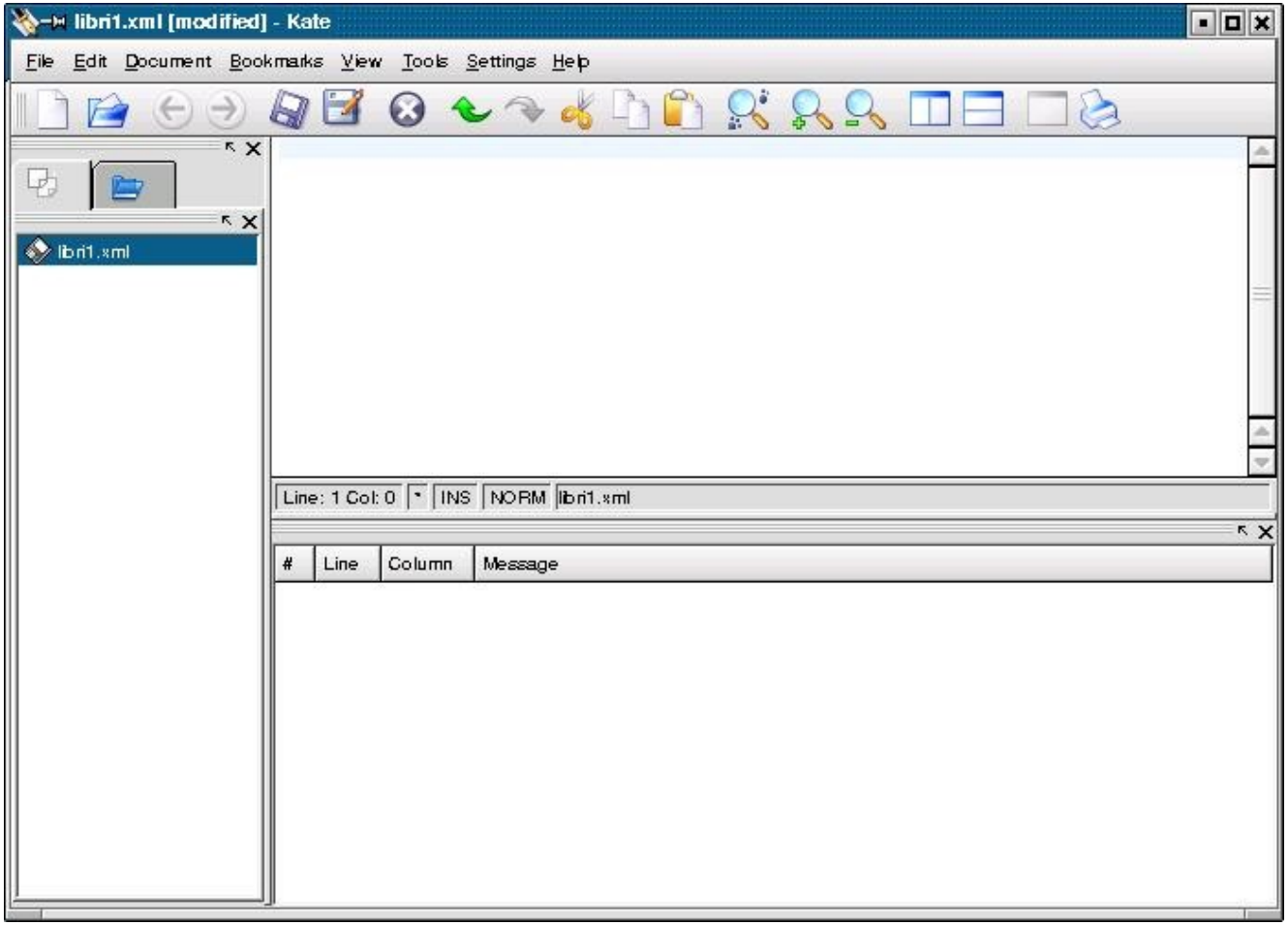
Esempio: file libri2.xml

http://validator.w3.org/

```
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<!DOCTYPE elenco
[<!ELEMENT elenco (libro+)>
<!ELEMENT libro (titolo,prezzo?)>
<!ELEMENT titolo (#PCDATA)>
<!ELEMENT prezzo (#PCDATA)>
]>
<elenco>
        <libro>
                <titolo> XML Guida di riferimento </titolo>
                <prezzo> 35 </prezzo>
        </libro>
        <libro>
                <titolo> Basi di dati </titolo>
                <titolo> Basi di dati2 </titolo>
        </libro>
</elenco>
```

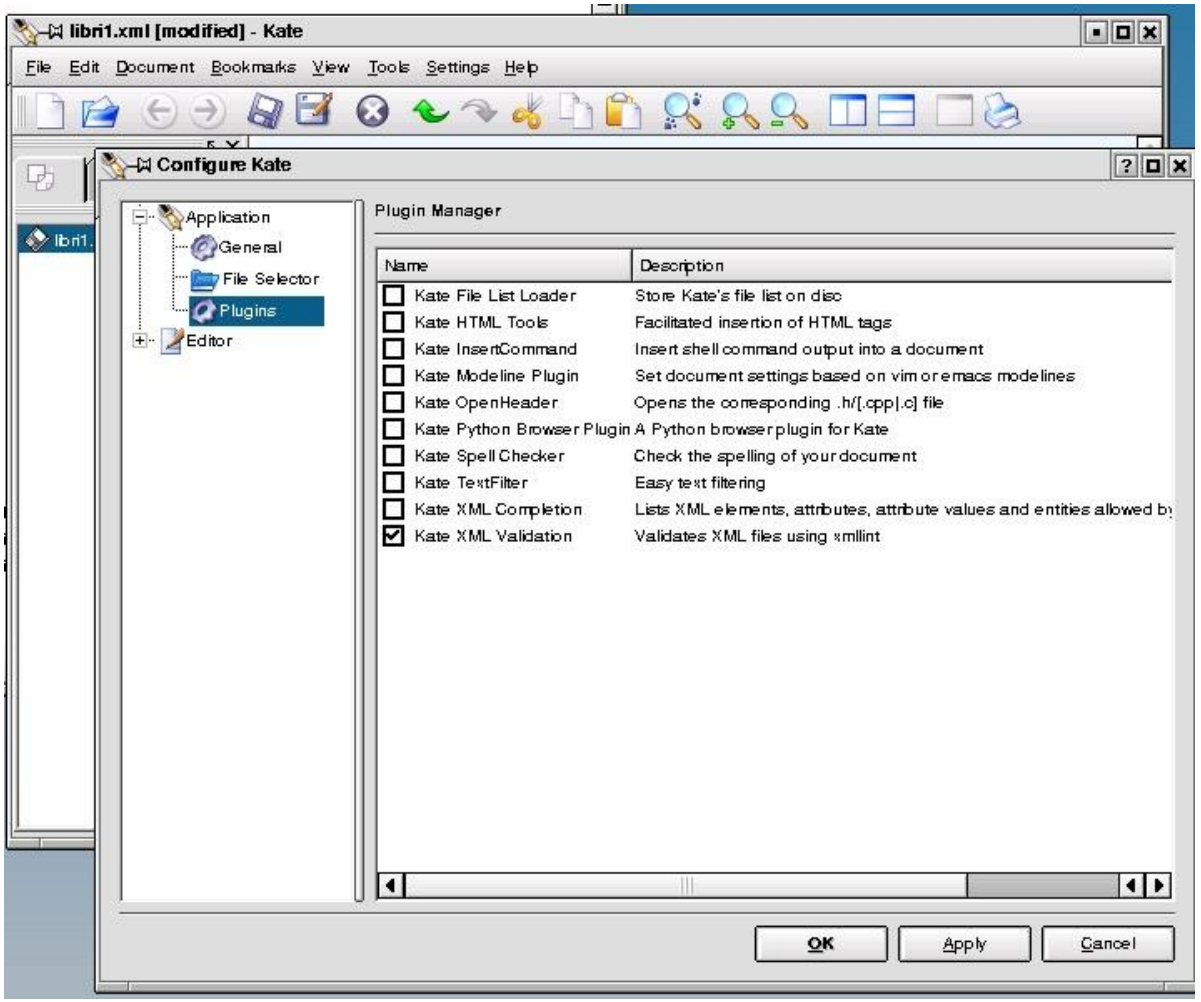
Kate (1)

- Lanciare kate



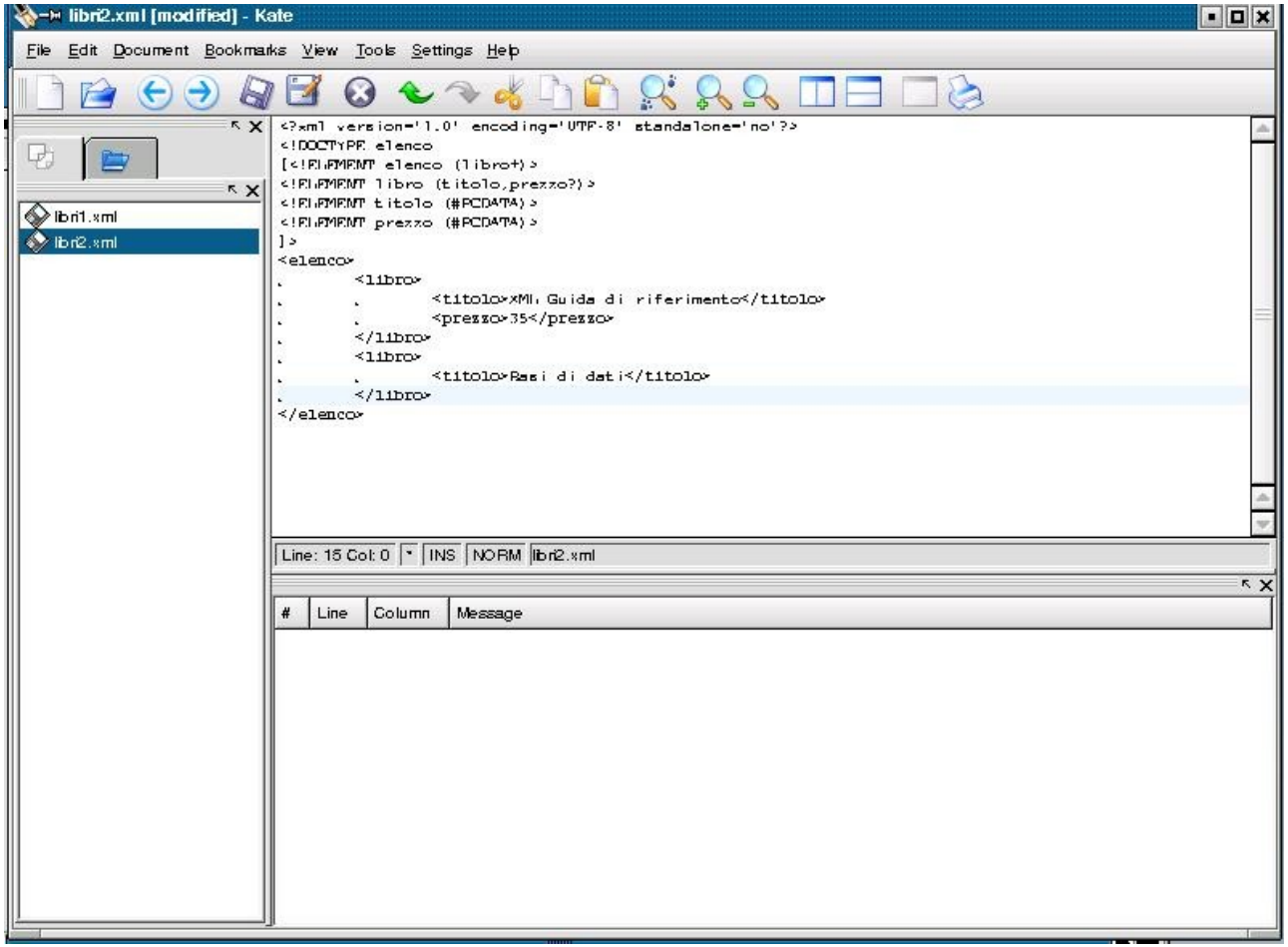
Kate (2)

- Settings
- Configure Kate
- Application
- Plugins
- Kate XML Validation
- OK



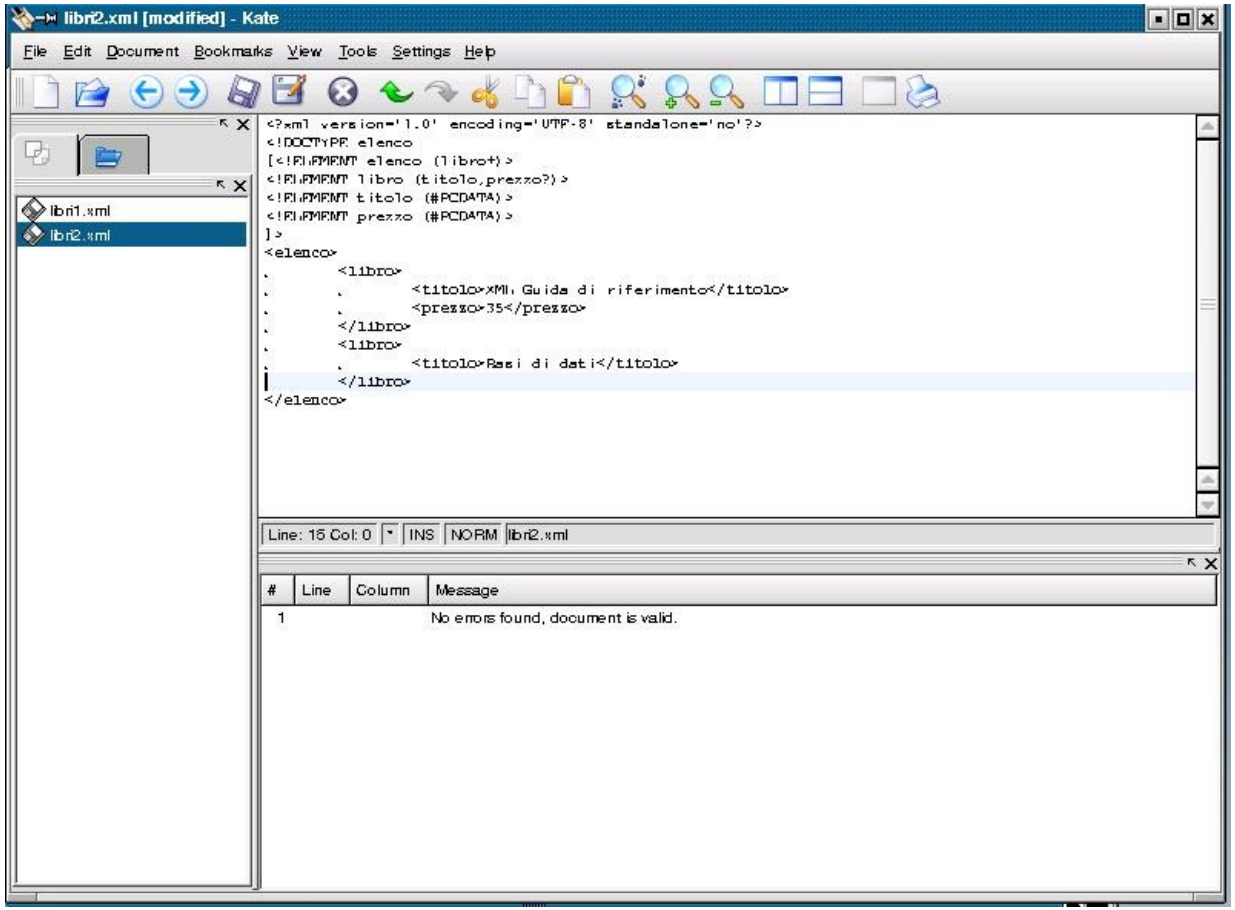
Kate (3)

- Creare o caricare il file XML



Kate (4)

- Selezionare Tools
- Selezionare Validate XML



Riferimenti

- Essential XML - Oltre il Markup

  Box Don, Skonnard Aaron, Lam John.

  Addison Wesley

Limiti di HTML

- In una pagina HTML non si possono gestire gli aspetti:
  - temporali: non esiste un sistema di coordinate temporali
  - spaziali: non esiste un sistema di coordinate spaziali che consenta di collocare gli oggetti nella pagina
  - sincronizzazione di file multimediali:
    - sono caratterizzati da un intervallo di tempo $[t_i,t_f]$
    - anche testo e immagini possono essere caratterizzati da un intervallo, se la loro permanenza nella finestra è prevista in un certo periodo

SMIL (Synchronized Multimedia Integration Language)

- SMIL è un linguaggio di integrazione e sincronizzazione per i file multimediali che consente di gestire tempo e spazio

- SMIL è basato su XML

- E' stato sviluppato dal W3C e rilasciato:
  - Nel 1998, la versione 1.0 (REC-smil-19980615).
  - Nel 2001 la versione 2.0 (REC-smil20-20010807).

Struttura del documento SMIL (1)

- Struttura generale (file.smil):
  <smil> intestazione + corpo </smil>

- Intestazione: <head> … </head>
  contiene definizioni e metainformazioni

- Corpo: <body> … </body>
  contiene i tag di sincronizzazione dei media

NB. SMIL è case-sensitive: i tag vanno scritti in minuscolo

Struttura del documento SMIL (2)

```
<smil>
   <head>
        <layout>
        </layout>
   </head>
   <body>
   </body>
</smil>
```

Struttura del documento SMIL (3)

- L'intestazione definisce:
  - Layout: spazio grafico
    - Aree spaziali in cui è divisa la presentazione (regioni)
  - Meta Tag
    - <meta name="title" content="esempio SMIL" />
    - <meta name="author" content="Luca Rossi" />
- Il corpo definisce:
  - I media contenuti nella presentazione
  - La sincronizzazione tra questi media

Layout

- Il layout è lo spazio grafico nel quale verranno sincronizzati i dati multimediali

- Si definisce con i tag

  <layout> </layout>

- Nel layout vanno definiti:
  - un root-layout che definisce lo sfondo
  - le region in cui saranno contenuti gli elementi multimediali

Root-layout

- Permette di definire lo sfondo

- Caratterizzato dagli attributi:
  - height (altezza)
  - width (larghezza)

Esempio:

```
<layout>
   <root-layout width="400" height="200" background-color="white"/>
</layout>
```

Region (1)

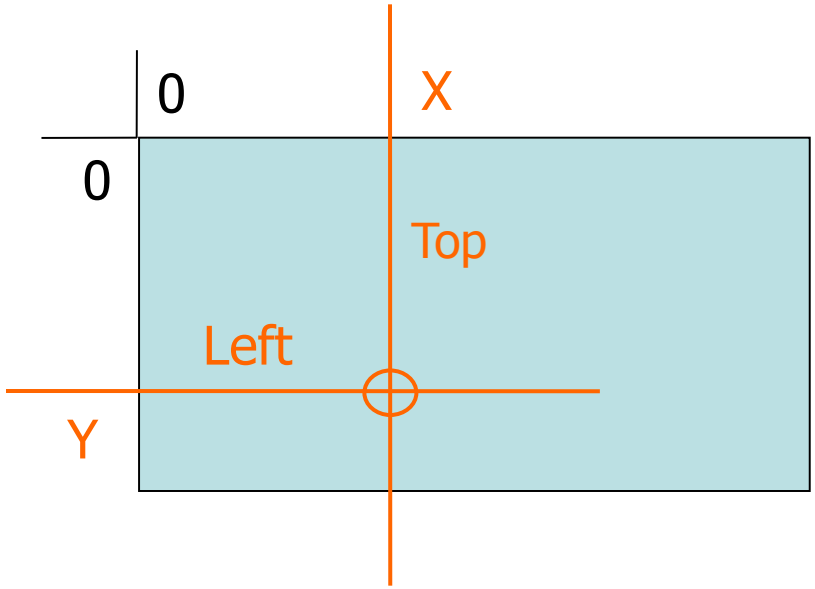
- Contengono gli elementi multimediali, la cui posizione viene specificata per mezzo di coordinate cartesiane
- Si definiscono con i tag

  `<region>  </region>`
- Caratterizzate dagli attributi:
  - `id`: specifica un identificatore univoco per la region
  - `left`: coordinata X
  - `top`: coordinata Y
  - `width`: larghezza dello spazio occupato
  - `height`: altezza dello spazio occupato

Region (2)

- Le coordinate devono essere calcolate a partire dall'angolo in alto a sinistra (sono descritte in termini di pixel)

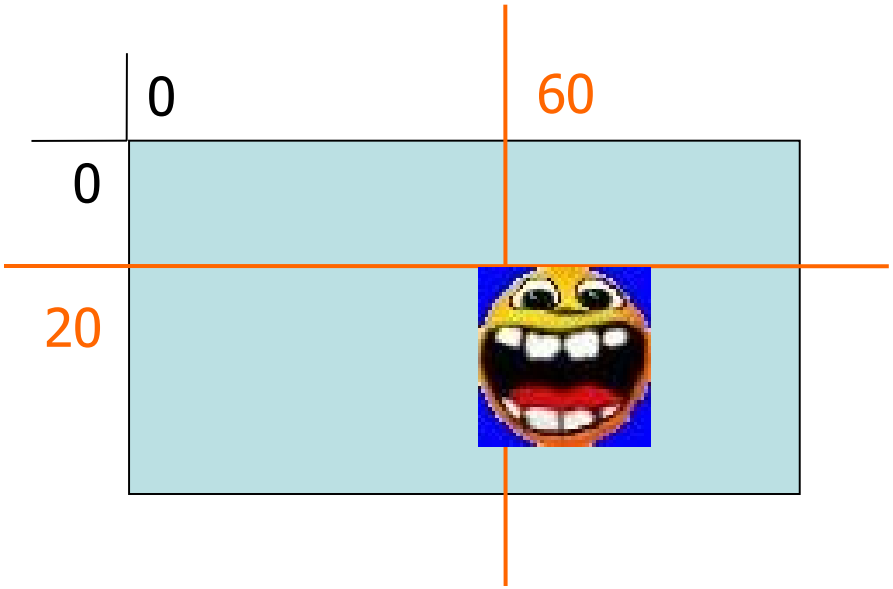
Esempio:



Region (3)

Esempio:

<region id="smile" left="60" top="20" width="32"
height="32" />



Esempio di intestazione

```
<smil>
    <head>
        <meta name="title" content="esempio" />
        <meta name="author" content="Luca" />
        <layout>
            <root-layout width="500" height="400"
                         background-color="white" />
            <region id="smile" left="60" top="20"
                    width="32" height="32" />
        </layout>
        ...
    </head>
    ...
</smil>
```

Region (4)

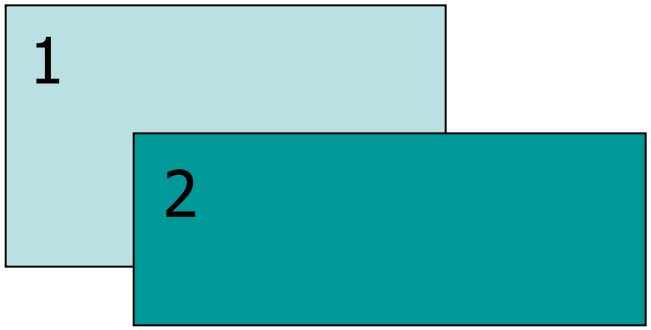
- Una <region> indica solo uno spazio identificato da un certo id
- La <region> non indica il reale contenuto dello spazio
  - Per identificare il contenuto della <region> nel corpo del documento dobbiamo associare un file
- Alle <region> possono essere associati diversi tipi di file multimediali:
  - Immagini: <img>
  - Testo: <text>
  - Video: <video>

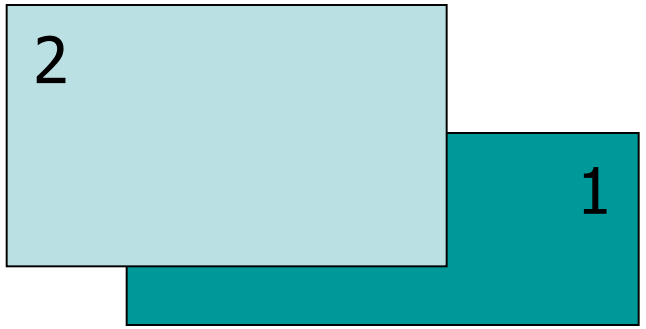
Esempio

```
<smil>
    <head>
        <layout>
            <root-layout width="500" height="400"  />
            <region id="smile" left="60" top="20"
                width="32" height="32" />
        </layout>
    </head>
    <body>
        <img src="smile8.gif" region="smile" />
    </body>
</smil>
```

Region (5)

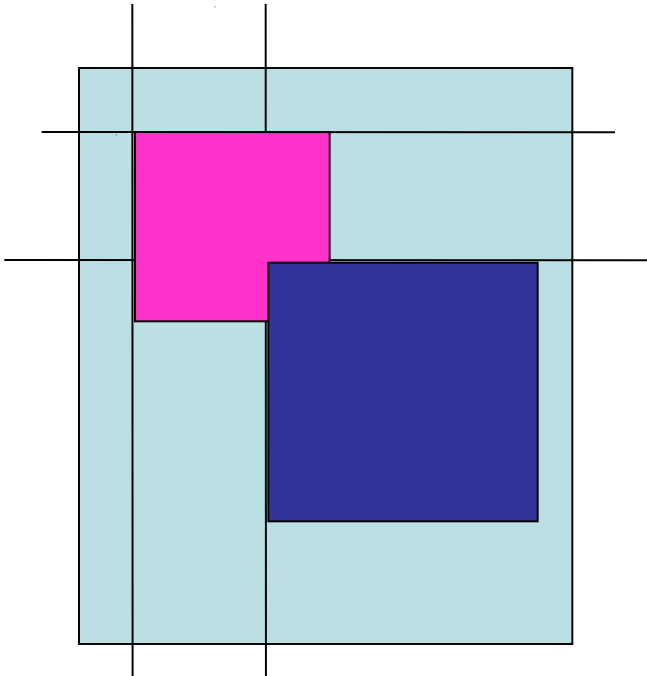
- Le regioni possono sovrapporsi
- Dobbiamo definire un ordine in modo da capire quale sovrascrive le altre
- L'ordine è definito dall'attributo z-index:
  - La regione con z-index maggiore sovrascrive le altre





Esempio

```
<smil>
    <head>
        <layout>
            <root-layout width="80" height="90" />
            <region id="region_1" left="30" top="30"
                width="40" height="40" z-index="2"/>
            <region id="region_2" left="10" top="10"
                width="30" height="30" z-index="1"/>
        </layout>
    </ head>
    <body>
        <img src="box1.gif" region="region_1" />
        <img src="box2.gif" region="region_2" />
    </body>
</smil>
```



L’attributo Fit

- Quando un oggetto multimediale (mm) non riempie esattamente l’area definita dalla <region> si può usare l’attributo fit
- Il tag fit può assumere diversi valori:
  - fill: riempie l’area distorcendo
  - hidden: vengono mantenute le dimensioni dell’oggetto mm
  - meet: regola l’oggetto mm mantenendo le proporzioni e riempie parte dell’area senza distorsioni
  - scroll: se necessario mette le barre di scorrimento
  - slice: regola l’oggetto mm mantenendo le proporzioni e riempie tutta l’area senza distorsioni. Parte dell’oggetto potrebbe venir tagliata
  - Il valore di default di "fit" è "hidden”

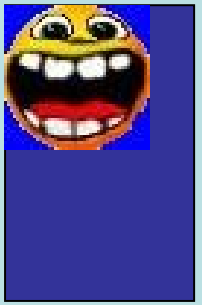
Esempio:

```
<region id="region_1" left=“30” top=“30” width=“40” height=“40”
        fit=“fill”/>
```

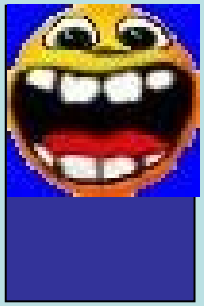
Esempio

Fill

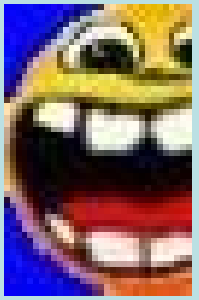
Hidden



Meet



Slice

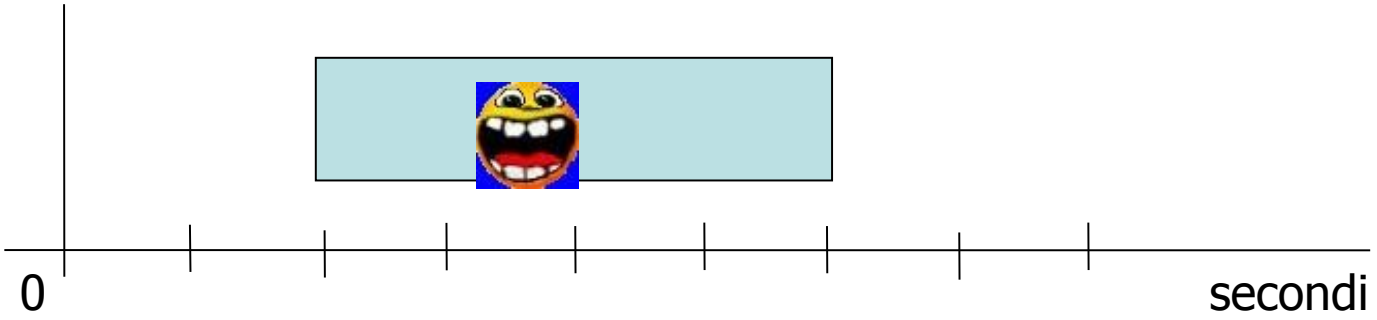


Il tempo

- Sui singoli tag relativi agli oggetti mm si possono aggiungere attributi che regolano la temporizzazione:
  - dur: indica la durata del media
  - begin: indica il momento di inizio
  - end: indica il momento di fine

Esempio:

```
<img src="smile8.gif" region="smile" dur="4s" begin="2s"/>
```

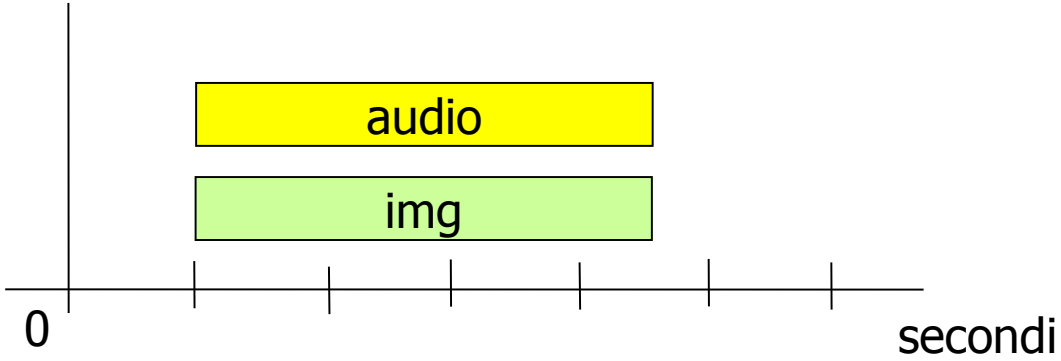


Sincronizzazioni di base: <PAR>

- <par> consente di riprodurre i media in parallelo
- Gli elementi figli di un elemento <par> possono sovrapporsi nel tempo
- L'ordine testuale dell'apparizione dei figli in <par> non è significativa per il tempo della presentazione

Esempio:

```
<par>
    <img src="box1.gif" region="region_1" />
    <audio src="helloo.wav"/>
</par>
```

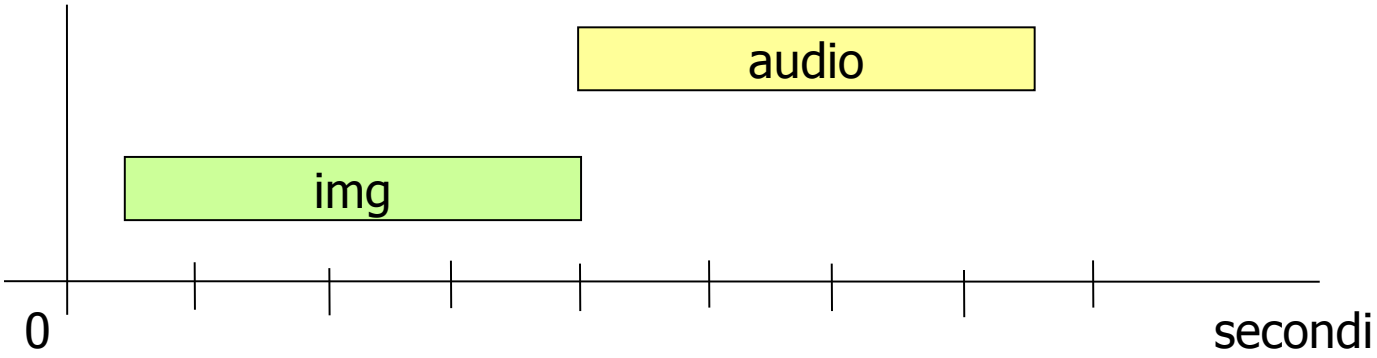


Sincronizzazioni di base: <seq>

- <seq> consente di riprodurre i media in sequenza
- I figli di un elemento <seq> formano una sequenza temporale

Esempio:

```
<seq>
    <img src="box1.gif" region="region_1" />
    <audio src="helloo.wav"/>
</seq>
```



Esempio

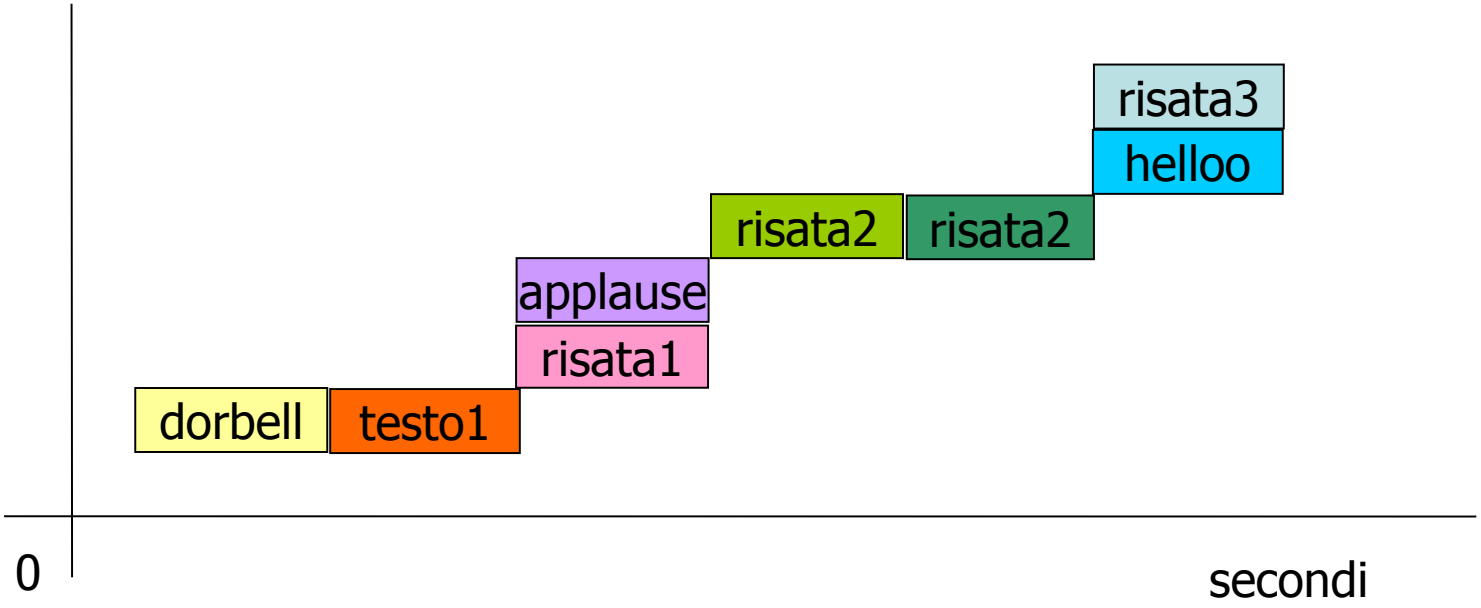
```
<smil>
    <head>
        <layout>
                ...
        </layout>
    </ head>
    <body>
        <par>
                <img src="box1.gif" region="region_1" dur="4s"/>
                <img src="box2.gif" region="region_2" dur="4s"/>
        </par>
        <seq>
                <img src="box1.gif" region="region_1" dur="4s"/>
                <img src="box2.gif" region="region_2" dur="4s"/>
        </seq>

    </ body>
</ smil>
```

Esempio

```
<smil>
    <head>
        <meta name="title" content="Esempio" />
        <meta name="author" content="Luca" />
        <layout>
            <root-layout width="250" height="250" background-color="white"/>
            <region id="region_1" left="10" top="10" width="400" height="80" />
            <region id="region_2" left="10" top="30" width="40" height="40" />
            <region id="region_3" left="10" top="70" width="80" height="80" fit="hidden"/>
            <region id="region_4" left="10" top="70" width="80" height="80" fit="fill"/>
        </layout>
    </head>
    <body>
        <seq>
            <audio src="doorbell.wav"/>
            <text src="testo1.txt" region="region_1" dur="5s"/>
            <par>
                <img src="risata1.gif" region="region_2"/>
                <audio src="applause.wav" />
            </par>
            <img src="risata2.jpg" region="region_3" dur="5s"/>
            <img src="risata2.jpg" region="region_4" dur="5s"/>
            <par>
                <audio src="helloo.wav"/>
                <img src="risata3.jpg" region="region_4" />
            </par>
        </seq>
    </body>
</smil>
```

Esempio: Timeline



Riferimenti

- W3C. The Synchronized Multimedia Integration Language (SMIL)

  http://www.w3.org/AudioVideo/