Introduction to XML & MusicXML

craig@ccrma.stanford.edu
20 January 2015

XML Development

- eXtensible Markup Language

Version 0 :: 1996
Version 1.0 :: 1998
Version 1.1 :: 2004
Version 1.1.5 :: 2008

XHTML 1.0 2000
1.1 2001
XHTML5 2008

http://en.wikipedia.org/wiki/XML

- Predecessor: SGML (Standardized Generalized Markup Language)

1970's – 1980's

HTML 1.0 1991
2.0 1995
4.0 1997
5.0 2008

http://en.wikipedia.org/wiki/Standard_Generalized_Markup_Language

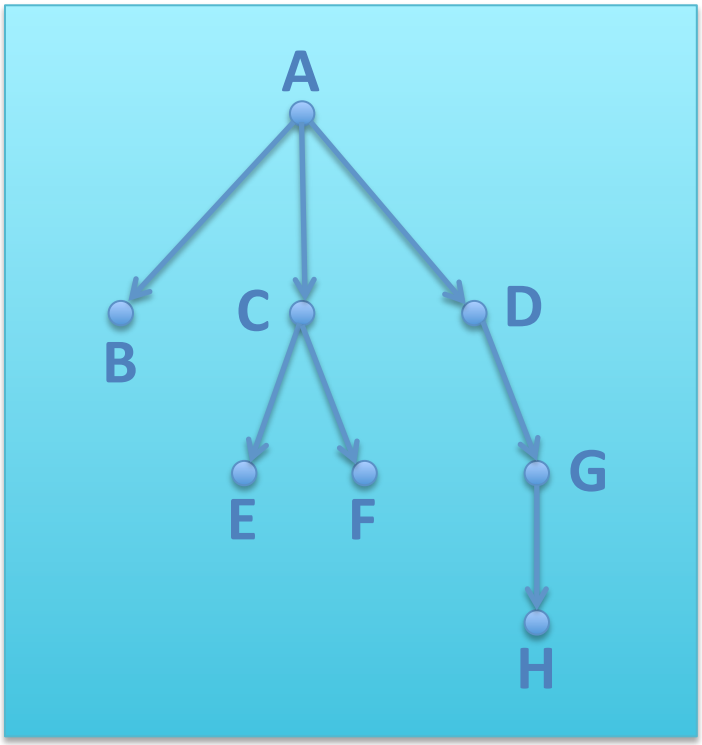
- Predecessor: GML (Generalize Markup Language)

1960's

http://en.wikipedia.org/wiki/IBM_Generalized_Markup_Language

XML data structure

- XML describes a tree structure:



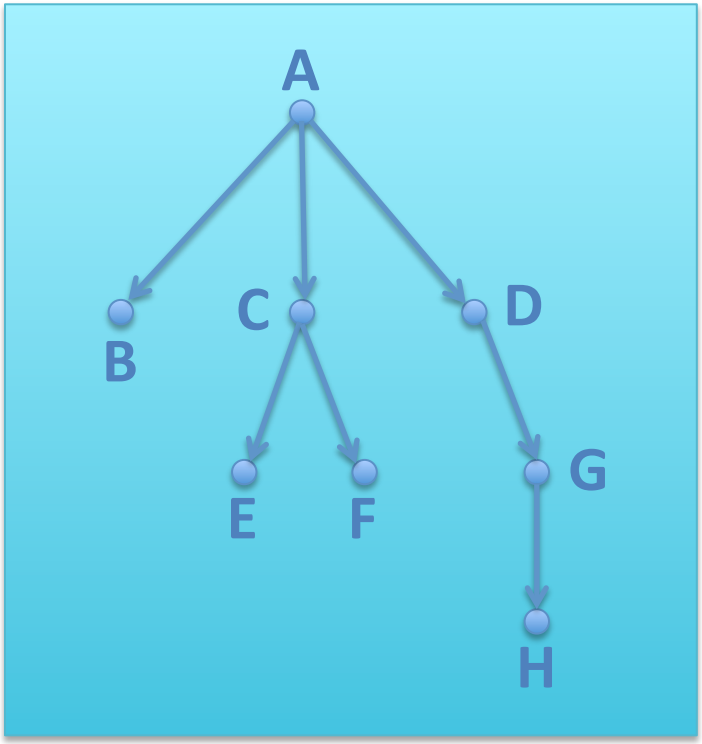
- Serialization:

```
<A>
   <B/>
   <C>
      <E/>
      <F/>
   </C>
   <D>
      <G>
         <H/>
      </G>
   </D>
</A>
```

- Equivalent serialization: <A><B/><C><E/><F/></C><D><G><H/></G></D></A>

XML data structure

- XML describes a tree structure:



- Same data structure as directories/folders on a hard disk

- Same conceptualization as LISP code:

(A B (C E F) (D (G (H))))

XML Terminology

```
<A>
   <B/>
   <C>
      <E/>
      <F/>
   </C>
   <D>
      <G>
         <H/>
      </G>
   </D>
</A>
```

- <C>…</C> is an ***element*** (tree node)
- C is the element's ***name***
- <C> is a *start* ***tag***
- </C> is an end tag
- <E/> and <F/> are *element* ***content*** of <C>
- Plain text inside of an element is *text* ***content***

- <H/> is an element without contents (terminal node)
- <H/> is equivalent to <H></H>
- Start tags must be followed by matching end tag, or the shorthand <xxx/> must be used.

Element Attributes

- Elements can contain a list of attributes within the start tag

<A a="1" b="two" c="1 and 2">

- Element **A** has three *attributes*: a, b, and c.
- A is the *name* of the attribute, 1 is its value.
- Attributes must have values. c="" represents an attribute without a value.
- Attributes are optional (similar to key values in LISP).
- The value of a is 1, the value of b is two and the value of c is 1 and 2.
- XML Attribute values *must* be enclosed in double or single quotes.
- Only one attribute of a given name allowed. Bad example: <A a="1" a="2">
- Attributes are considered unordered:

  <A a="1" b="two"> is identical to <A b="two" a="1">

HTML attributes do not need to be enclosed in quotes:
<table cellpadding=10> is equivalent to <table cellpadding="10">
XHTML is does not allow the first case since quotes are always needed.

Elements vs. Attributes

- Elements can contain subelements
- Attributes cannot contain subattributes

- Two similar (but not identical) ways of expressing the same data:

```
<A a="1" b="two" c="1 and 2"/>
```

```
<A>
  <a>1</a>
  <b>two</b>
  <c>1 and 2</c>
</A>
```

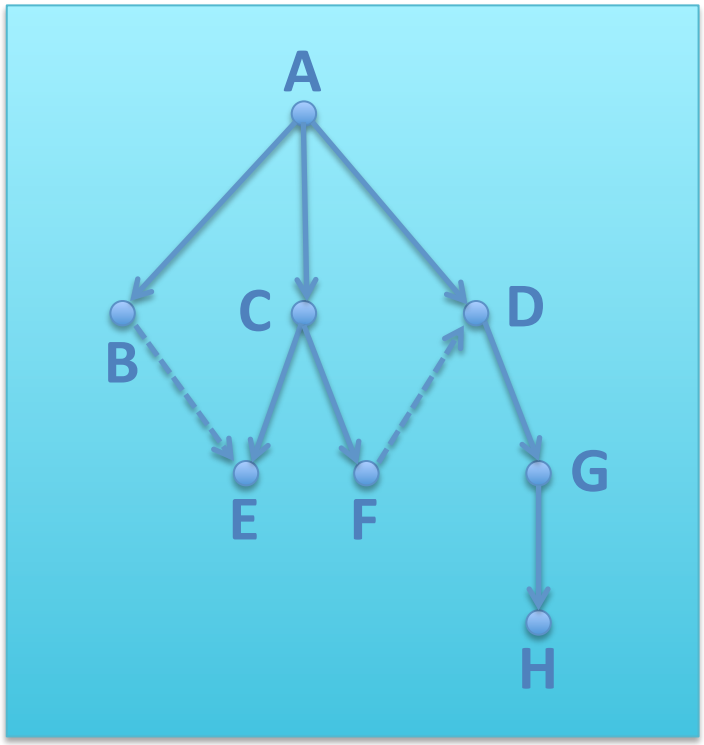
Informal shorthand for attribute a of element A (but not in data):

A@a

- Attribute a in the first example cannot be expanded later into subattributes
- Element a in the second example can be expanded later to include element contents

XML for non-tree structured data

- non-tree data can be shoe-horned into XML data structure



- Tree-like portions encoded as XML elements
- Non-tree connections handled by specialized id/idref/idrefs attributes.

```
<A>
   <B idref="e"/>
   <C>
      <E id="e"/>
      <F idref="d"/>
   </C>
   <D id="d">
      <G>
         <H/>
      </G>
   </D>
</A>
```

DTD:

```
<!ATTLIST B
   id      ID      #IMPLIED
   idref   IDREF   #IMPLIED>
```

- Similar to pointers in C.

XML declaration

- Used to indicate that the following data is XML data
- First characters in file must be “<?xml” (see UTF-16 below).

<?xml version=“1.0” encoding=“UTF-8” standalone=“yes”?>

Three attributes which *must* be in this order (but optional):

@version = version of XML being used (1.0 or 1.1).
@encoding = character set being used in data. (also UTF-16 which requires two endian bytes before opening <?)
* UTF-8 is backwards compatible with 7-bit ASCII
* UTF-16 is not.
@standalone = “yes” if no external definition file, “no” if DTD (Document Type Definition).

XML complete data file

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<A>
    <B idref="e"/>
    <C>
        <E id="e"/>
        <F idref="d"/>
    </C>
    <D id="d">
        <G>
            <H/>
        </G>
    </D>
</A>
```

Even more complete data file

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE A [
  <!ELEMENT A (B,C,D)>
  <!ELEMENT C (E,F)>
  <!ELEMENT D (G)>
   <!ELEMENT G (H)>
   <!ATTLIST B idref IDREF #IMPLIED>
   <!ATTLIST E id    ID    #IMPLIED>
   <!ATTLIST D id    ID    #IMPLIED>
]>
<A>
    <B idref="e"/>
    <C>
        <E id="e"/>
        <F idref="d"/>
    </C>
    <D id="d">
        <G>
            <H/>
        </G>
    </D>
</A>
```

Element A can have subelements B, C & D.

Element B can have an attribute named idref which can be set to a value which is the type IDREF.

Data/Structure definition separation

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE A SYSTEM "tree.dtd">
or <!DOCTYPE A SYSTEM "http://somewhere.com/tree.dtd">
or <!DOCTYPE A PUBLIC "-//Owner/Class Description//Language//Version" "tree.dtd">
<A>
    <B idref="e"/>
    <C>
        <E id="e"/>
        <F idref="d"/>
    </C>
    <D id="d">
        <G>
            <H/>
        </G>
    </D>
</A>
```

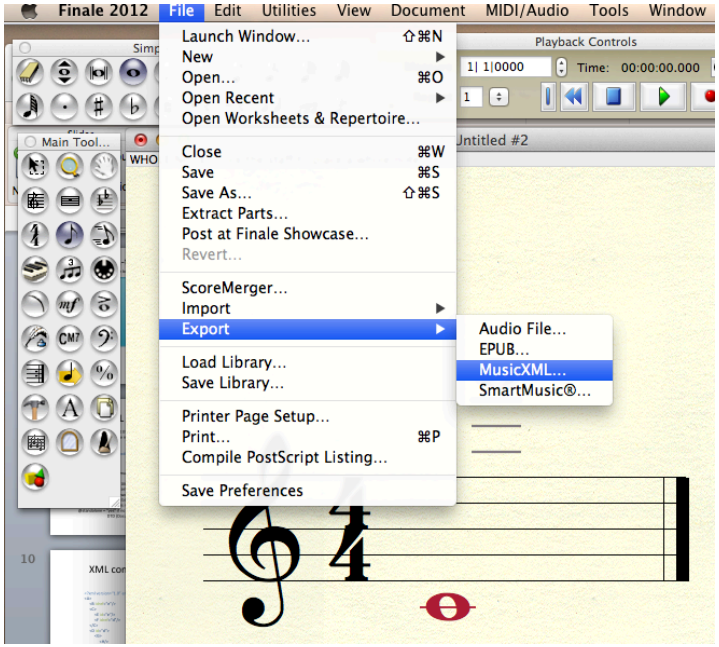
Formal Public Identifier

tree.dtd:

```
<!ELEMENT A (B,C,D)>
<!ELEMENT C (E,F)>
<!ELEMENT D (G)>
<!ELEMENT G (H)>
<!ATTLIST B idref IDREF #IMPLIED>
<!ATTLIST E id    ID    #IMPLIED>
<!ATTLIST D id    ID    #IMPLIED>
```

MusicXML

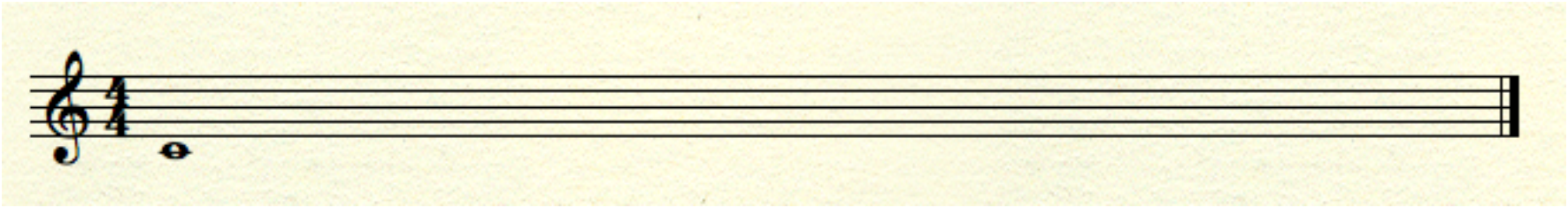
```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE score-partwise PUBLIC "-//Recordare//DTD MusicXML 1.0 Partwise//EN"
                "http://www.musicxml.org/dtds/1.0/partwise.dtd">
<score-partwise>
  <identification>
    <encoding>
      <software>Finale 2012 for Mac</software>
      <software>Dolet Light for Finale 2012</software>
      <encoding-date>2013-01-21</encoding-date>
    </encoding>
  </identification>
  <part-list>
    <score-part id="P1">
      <part-name>MusicXML Part</part-name>
      <score-instrument id="P1-I1">
        <instrument-name>Garritan: ARIA Player</instrument-name>
      </score-instrument>
      <midi-instrument id="P1-I1">
        <midi-channel>1</midi-channel>
        <midi-bank>15489</midi-bank>
        <midi-program>1</midi-program>
      </midi-instrument>
    </score-part>
  </part-list>
  <!--======================================================-->
```



<!-- ... --> is a comment in XML

visual barline for readability

MusicXML (2)



```
<part id="P1">
  <measure number="1">
   <print/>
   <attributes>
    <divisions>2</divisions>
    <key>
     <fifths>0</fifths>
     <mode>major</mode>
    </key>
    <time>
     <beats>4</beats>
     <beat-type>4</beat-type>
    </time>
    <clef>
     <sign>G</sign>
     <line>2</line>
    </clef>
   </attributes>
   <sound tempo="120"/>
```

divisions per quarter note

```
    <note default-x="86">
      <pitch>
       <step>C</step>
       <octave>4</octave>
      </pitch>
      <duration>8</duration>
      <voice>1</voice>
      <type>whole</type>
     </note>
     <barline location="right">
      <bar-style>light-heavy</bar-style>
     </barline>
    </measure>
  </part>
  <!--=========================================================-->
</score-partwise>
```

4 quarter notes

looks like a whole note

Compare to GUIDO:
[c/1]

(GUIDO content not separable from structure)