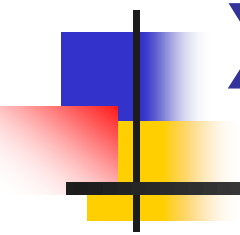
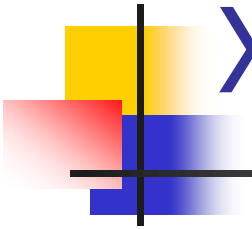


XML rakenteen suunnittelu

Jaana Holvikivi

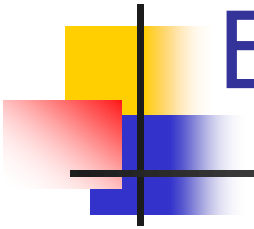


XML suunnittelu

Dokumentin ilmentymä

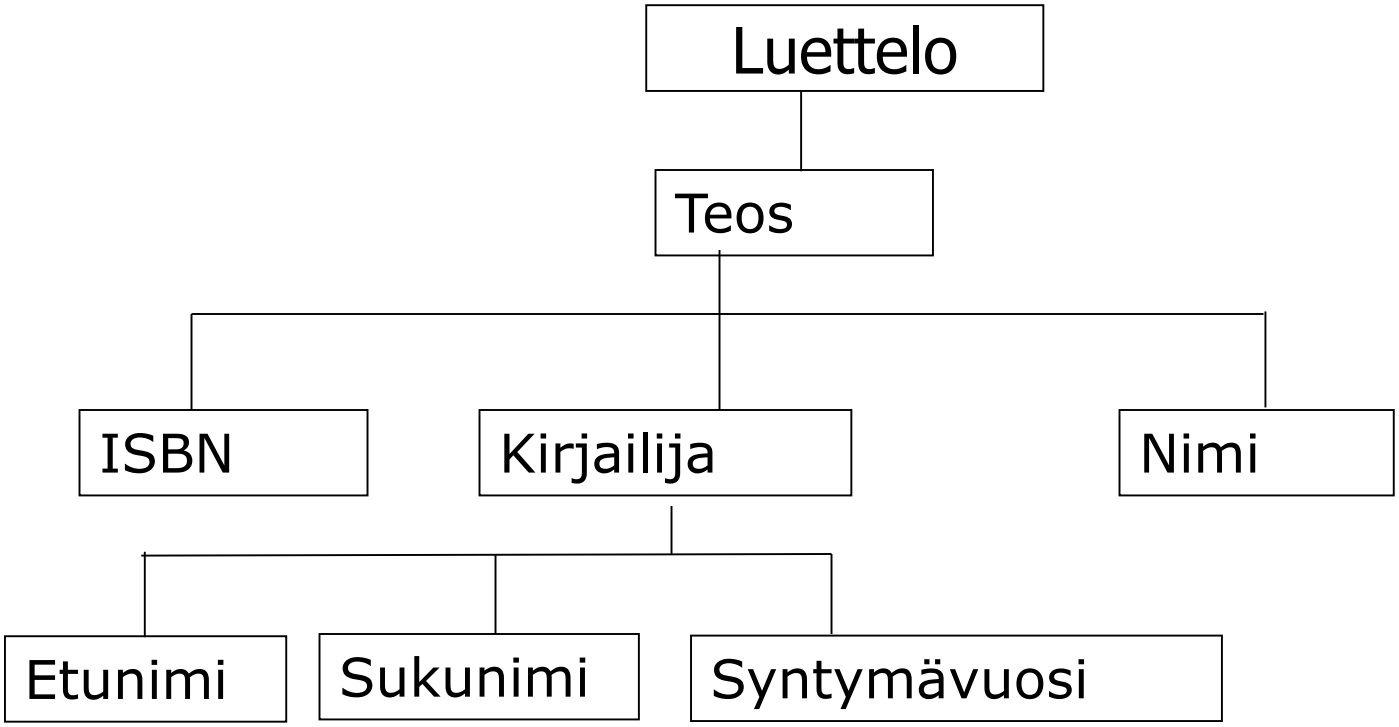
- elementit
- attribuutit (määritteet)
- entiteetit
- prosessointikäskyt

Elementtien sisäkkäisyys: säännöt

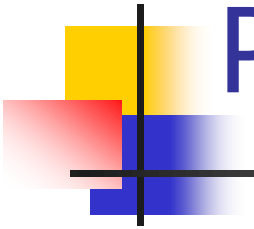


- aina p.o. vastaavat alku- ja lopputunnisteet
- elementtien nimien täytyy noudattaa XML:n sääntöjä
- elementin täytyy sisältyä täysin toiseen elementtiin (ei limittäin!)
- elementtihierarkia
  - juuri = dokumenttielementti - yksi ainoa!
  - Puumuotoinen rakenne
- **Hyvinmuodostuneisuus (= well-formed)**!

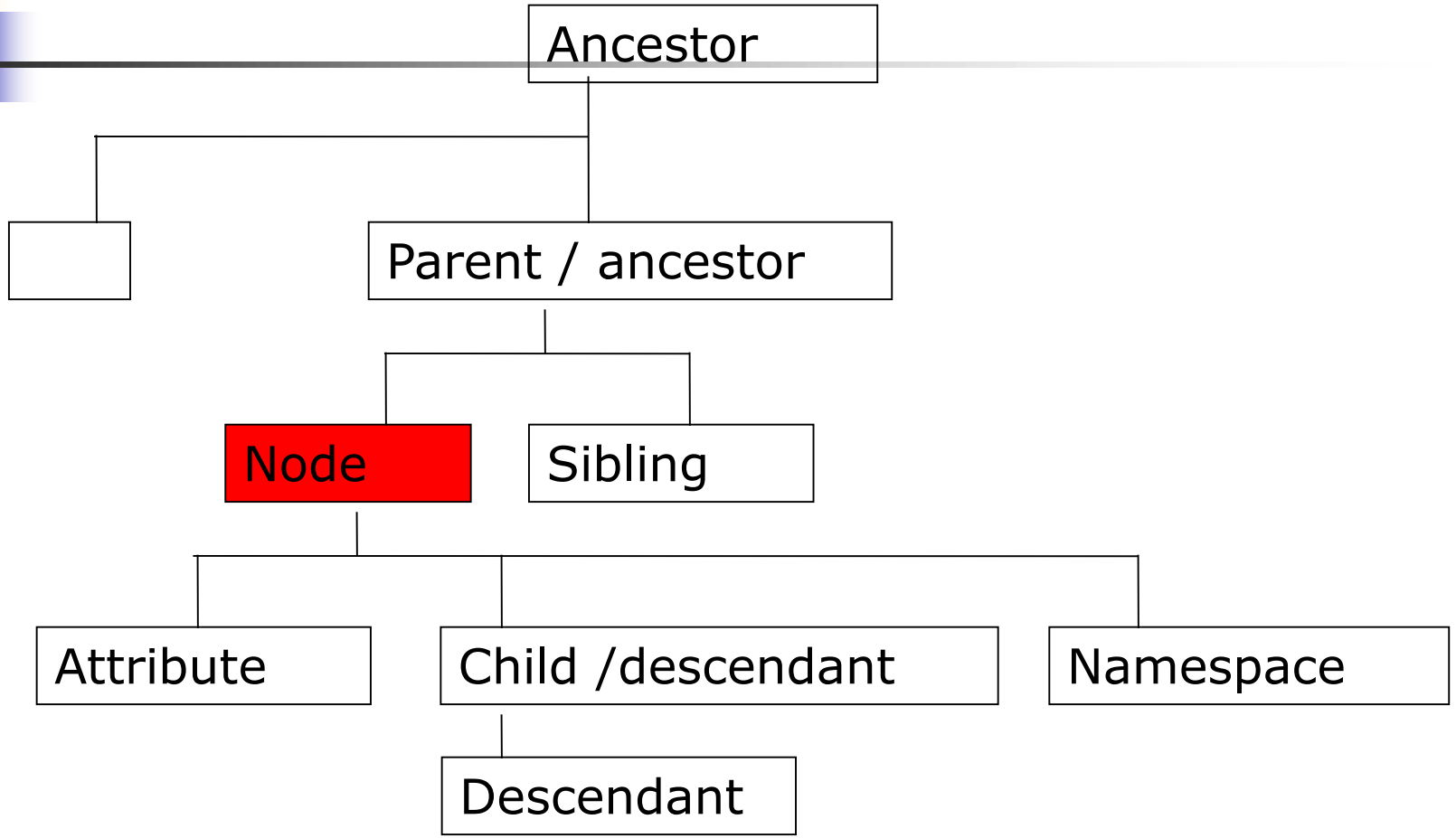
Dokumenttipuu



Puuterminologia



- Juuri, solmut, lehdet
- haarat
- vanhempi - lapset (äidit - tyttäret, isät - pojat)
- sisarukset
- esivanhemmat, jälkeläiset
- nimet

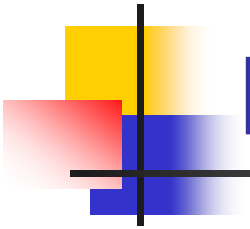

Ancestor
Parent / ancestor
Node
Sibling
Attribute
Child /descendant
Namespace
Descendant

Rakenteisuus (granulariteetti) 1

```
<country>
   <state>
            Washington,  Seattle
   </state>
   <state>
            Washington D.C., Washington
   </state>
</country>
```

Rakenteisuus 2

```
<country>
    <state>
      <state_name>Washington</state_name>
      <capital>Seattle</capital>
    </state>
      <state>
      <state_name>Washington D.C.</state_name>
      <capital>Washington</capital>
    </state>
</country>
```



Rakenteisuus 3

- mitä enemmän rakennetta sitä enemmän tunnisteita
- hieno vs. karkea
- hienorakenteinen - tarkempi
  - etsintä
  - muotoilu

Design: data vs rakenne

- Esimerkkejä:
  - http://users.metropolia.fi/~jaanah/ElDocCP/material/Pract2bmenu1.xml
  - http://users.metropolia.fi/~jaanah/ElDocCP/material/pract2bmenushort.xml

Erota data rakenteesta

```
<cafe>
<dishes>
        <dish type=”main” kind=”seafood”>
                <name>Seafoodplatter</name>
                <price>28</price>
        </dish>
        <dish type=”starter” kind=”vegetable”>
                <name>Redbeet salad</name>
                <price>8,50</price>
        </dish>
…
```

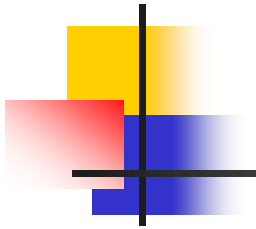
Attribuutit (määritteet)

- Elementin ominaisuuden tai sisällön tarkenne
- Liitetään alkutunnisteisiin (tai tyhjien elementtien tunnisteisiin)
  - attribuutin nimi ja arvo
- aina vain yksi arvo
- 'arvossa' voi olla mitä tahansa merkkejä

```
<book author=”Zadie Smith">
...
</book>

<text keywords="XML SGML">
...
</text>
```

Varatut attribuutit



- xml:lang
  - dokumentin kieli
  - ISO 639 (kielet) (+ ISO 3166, maat)
  - tai käyttäjän määrittelemä tai IANA
- xml:space
  - kuuluuko 'white space' tulosteeseen vai ei
  - arvot: preserve tai default

xml:lang

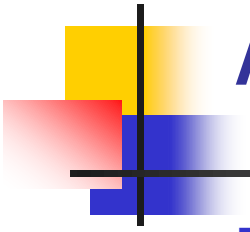
```
<product>
<paragraph xml:lang="en">
...
</paragraph>
<paragraph xml:lang="fi">
...
</paragraph>
</product>
```

xml:lang

```
<p xml:lang="en">
The quick brown fox jumps over the lazy dog.</p>

<p xml:lang="en-GB">What colour is it?</p>
<p xml:lang="en-US">What color is it?</p>

<sp who="Faust" desc='leise' xml:lang="de">
        <l>Habe nun, ach! Philosophie,</l>
        <l>durchaus studiert mit heißem Bemüh'n.</l>
</sp>
```



Attribuutti vai lapsielementti?

- attribuuteilla ei voi olla useampia kuin yksi arvo
- attribuutteja ei voi laajentaa, elementeille sen sijaan voidaan luoda lapsielementtejä
- attribuutin avulla ei ilmaista rakennetta
- attribuutteja käsitellään ohjelmakoodissa eri tavalla kuin elementtejä
- attribuuttiarvojen testaaminen DTD:n suhteen rajoittunutta

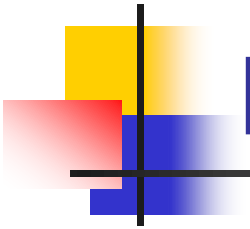
Merkkausesittelyt

- Sisältävät käskyjä XML-prosessorille
  - alku: <!
  - Loppu: >
  - dokumenttityypin rakenne, osien esittelyt, jne.

  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "DTD/xhtml1-strict.dtd">
- Kommentit

  <!-- Tämä on kommentti -->
- CDATA

  <![CDATA[tekstiä “4>2” ei &xml-prosessoida]]>



Entiteetit

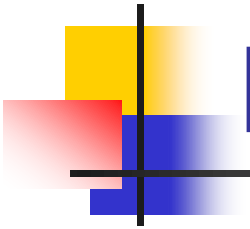
- Tietoyksikkö - nimetty ja itsenäinen osa, jota käytetään erilaisten sisältöjen säilyttämiseen.
- esim. duplikaatteja varten, tai
- suuren dokumentin jako pienempiin osiin
- sisäiset entiteetit
  - vakiomerkkijonot
- ulkoiset entiteetit
  - jäsennetty - korvaustekstit
  - binäärientiteetit - piirustukset, kuvat, ääni, video

Entiteettien käyttö

Jos esim. entiteetti **&metrop;** sisältää merkkijonon
"Metropolia ammattikorkeakoulu"

voimme kirjoittaa
Opiskelupaikkani on &metrop;.

- Älä käytä < tai >
- varatut entiteetit: < ja >
- tietyt prosessorit pystyvät käsittelemään
  - käyttäjä kirjoittaa esim. kaavassa < ja >
  - prosessori kääntää merkkijonoiksi < ja >



Entiteettien prosessointi

- XML-prosessorin tulee käsitellä entiteettejä oikein
  - korvata tekstiä
  - sisällyttää binäärientiteetit dokumenttiin
  - esittää entiteetin määrätyllä tavalla

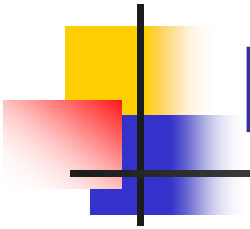
Prosessointikomennot

- Prosessointikomento on dokumenttiin liitetty komento tai ohje, jonka XML-jäsennin välittää dokumenttia käsittelevälle sovellukselle.
- erotusmerkit **<?** ja **?>**

- esimerkki: XML-esittely:

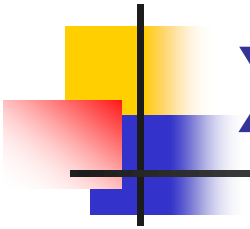
<?xml version="1.0" encoding="ISO-8859-1" standalone='yes'?>

  - versio on 1.0
  - merkistö
  - ei ulkoisia määrittelyjä
- <?xml-stylesheet type="text/xsl" href="center.xsl"?>



Hyvinmuodostuneisuus

- XML-dokumentti on **hyvinmuodostettu** (well-formed) jos se
  - sisältää täsmälleen yhden juurielementin ja muut mahdolliset elementit alkavat ja päättyvät saman elementin sisällä eli elementit ovat tasapainossa,
  - vastaa XML-määrittelyn asettamia hyvinmuodostetun dokumentin rajoituksia,
  - jokainen jäsennetty entiteetti on hyvinmuodostettu

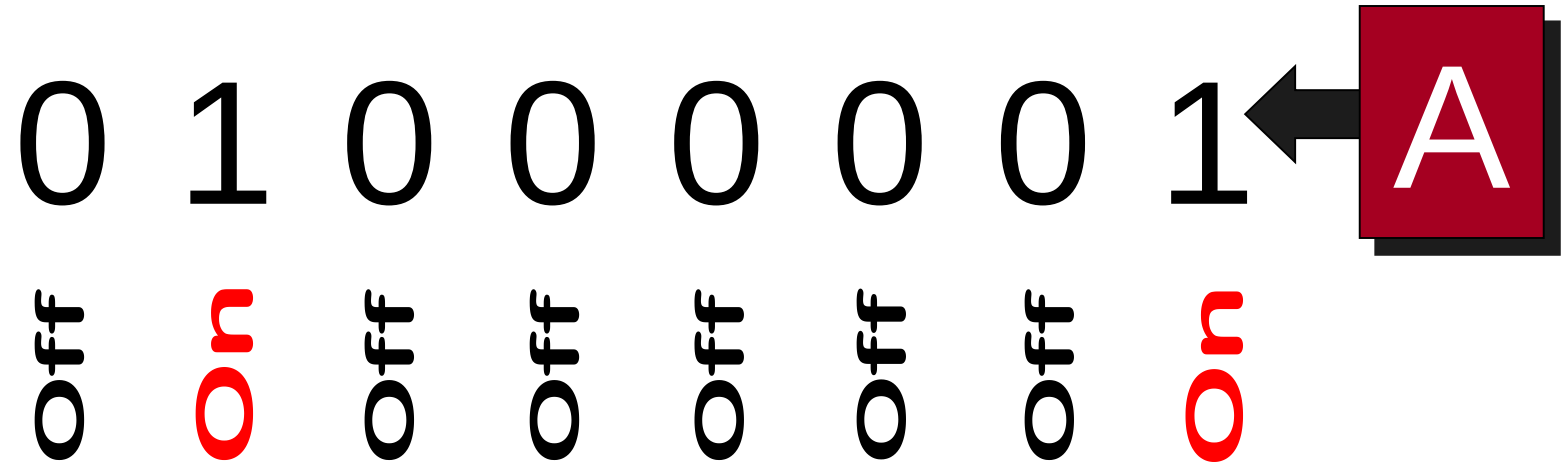


XML parseri eli prosessori

- pysähtyy virheeseen, ei yritä tulkita päinvastoin kuin HTML-selaimet
- IE:een sisältyy MSXML (uudemmat versiot paremmin standardia noudattavia)
- muissakin selaimissa parseri
- Apache projekti Xerces (Java, C++)
- Saxon
- Visual Studio .NET

Merkkikoodistot: esimerkki ASCII

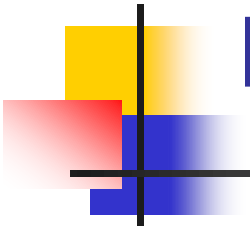
“A” esitetään bittijonona
1 tavu/ byte = 8 bittiä



ASCII -merkistö

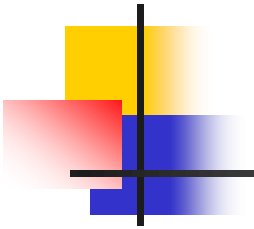
Character	ASCII Code	Character	ASCII Code
A	100 0001	0	011 0000
B	100 0010	1	011 0001
C	100 0011	2	011 0010
D	100 0100	3	011 0011
E	100 0101	4	011 0100
F	100 0110	5	011 0101
G	100 0111	6	011 0110
H	100 1000	7	011 0111
I	100 1001	8	011 1000
J	100 1010	9	011 1001
K	100 1011	Space	010 0000
L	100 1100	.	010 1110
M	100 1101	(	010 1000
N	100 1110	+	010 1011
O	100 1111	&	010 0110
P	101 0000	$	010 0100
Q	101 0001	*	010 1010
R	101 0010	)	010 1001
S	101 0011	;	011 1011
T	101 0100	,	010 1100
U	101 0101	-	101 1111
V	101 0110	?	011 1111
W	101 0111	:	011 1010
X	101 1000	=	011 1101
Y	101 1001		

Merkkikoodistot



- ASCII American Standard Code for Information Interchange
  - 7 bittiä $2^7 = 128$
  - 8 bittiä $2^8 = 256$
- ANSI, ISO Latin-1: 8 bittisiä
- Unicode (IBM, Microsoft, Sun)
  - 16 bittinen: $2^{16} = 65\,536$
  - http://www.unicode.org/charts/
- ISO 10646 neljä tavua

Tyhjät välit (white space)



- välilyönti, rivinvaihto, sarkaimet

character	Unicode value
tab	#x9
newline	#xA
carriage return	#xD
space	#x20

- rivinvaihto eri käyttöjärjestelmissä:

MacOS	CR
Unix	LF
Windows	CR LF

Tyhjät välit (white space) jatkuu

- HTML jättää näyttämättä
- XML: periaatteessa säilyttää
- normalisointi = ylimääräisen tyhjän poistaminen
  - elementtien välillä olevat tyhjät välit poistetaan
  - rivinvaihdot korvataan välilyönnillä
  - lohkon alusta ja lopusta poistetaan tyhjät välit
  - literaaliarvon ympäriltä poistetaan lainausmerkit
  - entiteettiviittaukset korvataan vastaavilla merkkijonoilla
- jos halutaan, että tyhjät välit säilyvät:
  - white-space: pre /* normal, nowrap */