ELEC-C8203 Automaatiojärjestelmät 2

XML Schema - Teoria

Pekka Aarnio

Sisältö

- Osa 1
  1. Mikä on XML-Schema-kieli
  2. Tietotyypit ja elementtien tyyppihierarkia
  3. Elementtien määrittely
  4. Simple type elementit
  5. Complex type elementit
  6. Attribuuttien määrittely
  7. Esimerkki: XML -scheman vaiheittainen määrittely

Luento 3: OSA1

XML-SCHEMA

1. Esimerkki – Miltä yksinkertainen XML Schema dokumentti näyttää?

- XML dokumentti 09-08.xml & XML Schema dokumentti 09-06.xsd

XML dokumentti 09-08.xml

```
<?xml version="1.0"?>
<wonder xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="09-06.xsd" >
  <name>Colossus of Rhodes</name>
  <location>Greece</location>
  <height>107</height>
</wonder>
```

XML Schema dokumentti 09-06.xsd

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="wonder">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="location" type="xs:string"/>
      <xs:element name="height" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

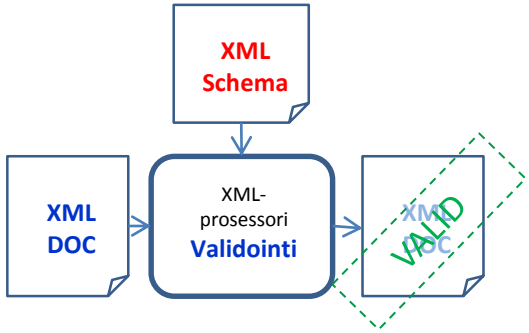
XML Schema

- XML Schema on kieli, jolla määritellään muita XML-pohjaisia kieliä
  - XML Schema -määrittelyillä luodaan kielen rakenteet
    - määritellään käytettävissä olevat elementit ja attribuutit
    - asetetaan rajoituksia elementtien sisäkkäisyydelle ja peräkkäisyydelle
    - määritetään attribuuttien arvoilla tyypit ja mahdolliset raja-arvot
    - lisätään attribuuteille oletusarvot tai todetaan tietyt attribuutit pakollisiksi
  - Kielen rakenteen pohjalta syntyy dokumentin sisältö ja semantiikka
- XML-dokumenttien kielenmukaisuus voidaan validoida kielen schemaa vasten
    - määritelty rakenne mahdollistaa dokumenttien tarkistuksen ja helpottaa dokumenttien koneellista käsittelyä

Katso myös: http://www.w3.org/XML/Schema#dev

Online validation tool: http://www.xpathtester.com/validate

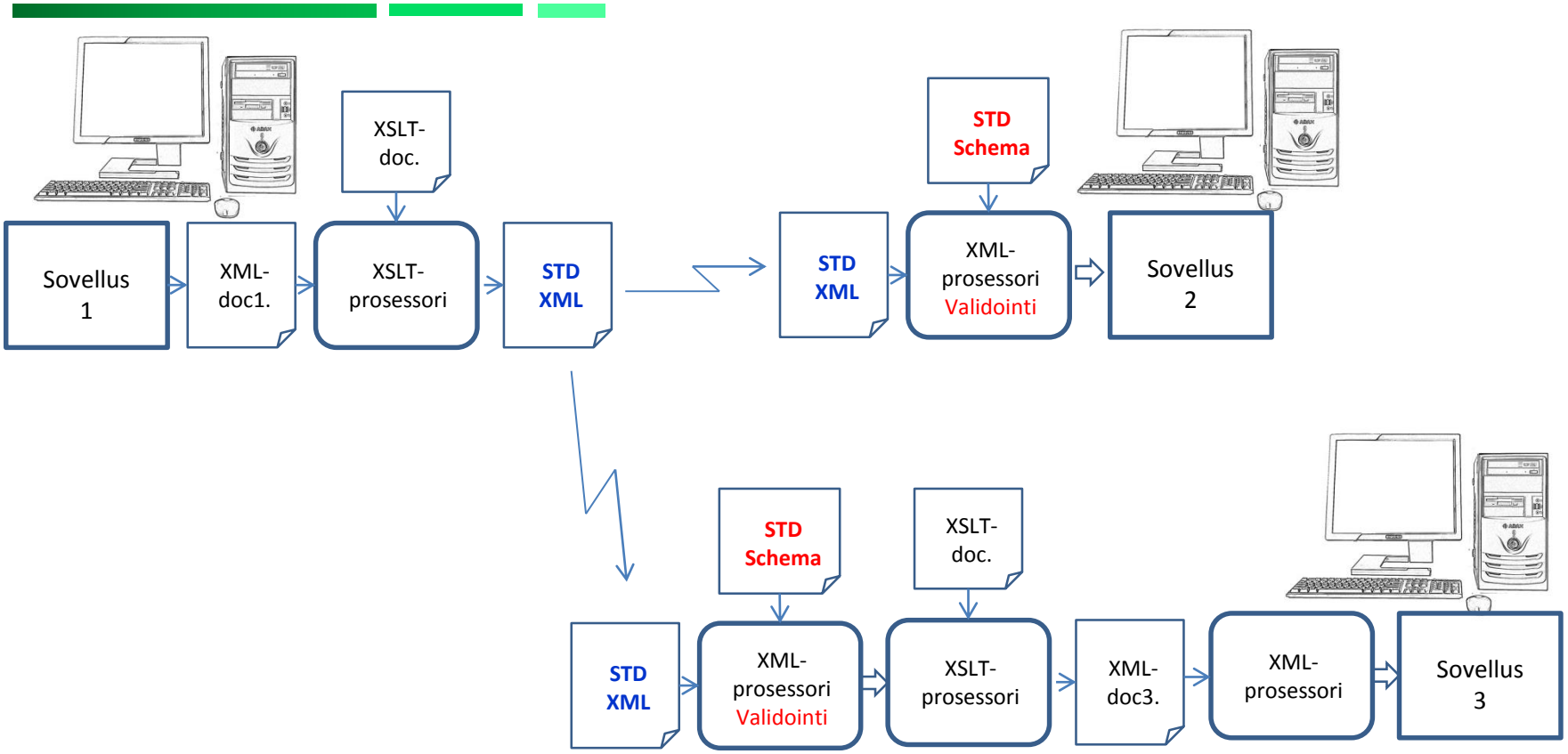
XML-dokumentin validointi



- Validoinnissa verrataan XML-dokumenttia sen XML Schemaan
  - dokumentin jokainen elementti, attribuutti ja tekstikenttä tarkastetaan
  - mikäli Schema ei salli jotain komponenttia tai jokin vaadittu komponentti puuttuu, dokumentti ei ole Scheman mukainen ja validointi epäonnistuu
- Validoinnilla sovellus voi varmistaa, että sisään luettu dokumentti on ainakin syntaktisesti oikein
  - validointi ei löydä semanttisia tai loogisia virheitä

HUOM: Validi dokumentti ei tarkoita samaa kuin oikeamuotoisuus (well formed). Kaikkien XML-dokumenttien tulee olla oikeamuotoisia. Kun taas validi XML-dokumentti on jonkin Scheman määrittelemän kieliopin mukainen.

Tietointegraatio – Standardoitu tiedonsiirto



XML Scheman ominaisuudet

- Käyttää XML:n syntaksia
  - toisin kuin DTD (Document Type Definition)*
- Tukee nimiavaruuksia
- Mahdollistaa tietotyyppien käytön
  - valmiiksi määritellyt
  - käyttäjän itse määrittelemät
- Mahdollistaa monimutkaisten ja uudelleenkäytettävien mallien rakentamisen
- Käyttää perintää
- Ei tue entiteettien määrittelyä
  - Tähän tarkoitukseen on käytettävä edelleen DTD:ta

*HUOM:
DTD:tä ei käsitellä tällä kurssilla.
[Goldberg, Ch.6]

XML-dokumentti <-> XML-schema

- XML-dokumentissa viitataan XML-Schemaan kahdella eri tavalla riippuen XML-Scheman määrittelyistä:
  - 1. tapa: jos XML-Schema määrittelee vain validin dokumentin rakennekieliopin:
    - *xsi:noNamespaceSchemaLocation*
  - 2. tapa*: jos XML-Schema määrittelee validin dokumentin rakenteen lisäksi myös kohdenimiavaruuden (targetNamespace), johon elementtien ja attribuuttien nimet määritellään kuuluviksi:
    - *xsi:schemaLocation*

*Tapa2: Nimiavaruuden määrittelevään XML-Schemaan viittaaminen esitellään tarkemmin luennon osassa B

XML-dokumentti <-> XML-schema – tapa1

- XML-dokumentissa viitataan validin dokumentin kieliopin määrittelevään XML-Schemaan
- Viittaus esitetään dokumentin juurielementin attribuutilla:
  - *xsi:noNamespaceSchemaLocation**
  - attribuutti kuuluu *XMLSchema-instance* nimiavaruuteen, joten ko. nimiavaruus täytyy myös määritellä juurielementissä
- XML-Schema-dokumentin juurielementti on *xs:schema*, joka kuuluu *XMLSchema*-nimiavaruuteen

*Huom: Nimiavaruuden määrittelevään XML-Schemaan viitataan toisella tavalla. Ks tarkemmin luennon osa B)

Dokument file: 09-06.xml

```
<?xml version="1.0"?>
<wonder xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="09-06.xsd" >
  <name>Colossus of Rhodes</name>
  <location>Greece</location>
  <height>107</height>
</wonder>
```

Schema file: 09-06.xsd

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="wonder">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="location" type="xs:string"/>
      <xs:element name="height" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

XML-Schema

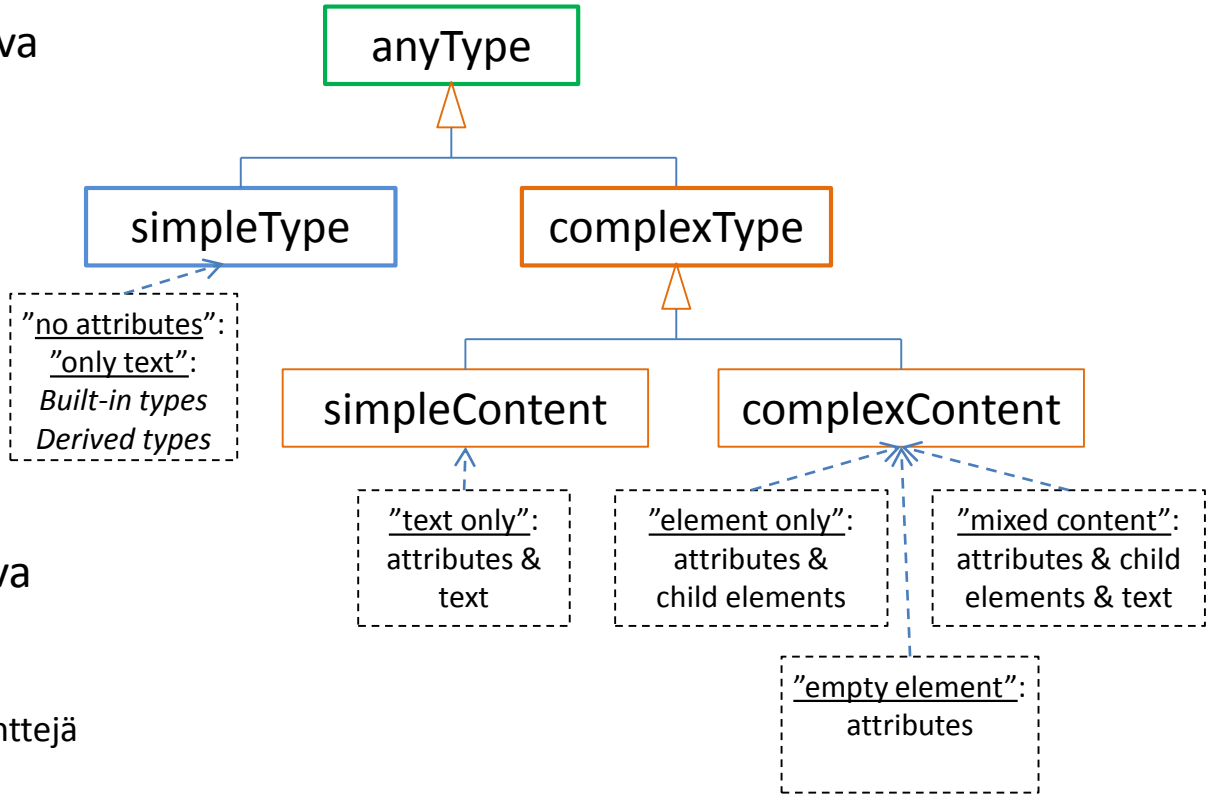
ELEMENTTIEN MÄÄRITTELY

Tietotyypit

- XML Schema määrittää kaksi tietotyyppikategoriaa
  - yksinkertainen (simpleType)
    - attribuuttien arvot
    - tekstikentät elementeissä
    - valmiiksi määritellyt (built-in) tyypit
    - käyttäjän omat johdetut tyypit (derived custom simple types)
  - monitahoinen (complexType)
    - lapsielementtejä ja/tai attribuutteja sisältävät elementit
    - ei ole valmiiksi määriteltyjä tyyppejä
    - kaikki monitahoiset tyypit täytyy johtaa
- XML-dokumentin elementit voivat olla joko yksinkertaisia tai monitahoisia

Elementtien tyyppihierarkia

- Yksinkertaista tyyppiä oleva elementti voi sisältää
  - Vain tekstiä
  - Mutta ei attribuutteja



- Monitahoista tyyppiä oleva elementti voi sisältää
  - Attribuutteja ja tekstiä
  - Attribuutteja ja lapsielementtejä
  - Attribuutteja
  - Attribuutteja ja lapsielementtejä ja tekstiä
  - Ei mitään

Elementtien määrittely

- Elementit määritellään scheman <xs:element>-elementillä:

  <xs:element name="document"/>

  - määriteltiin XML-dokumentin elementti: <document>
  - attribuuteilla voidaan lisätä määreitä elementtiin, esimerkiksi säätää elementin sisältö pelkäksi tekstiksi

  <xs:element name="document" type="string"/>

Paikallinen ja globaali tyyppi

- Elementin sisältö voidaan määrittää joko paikallisesti (local type) tai käyttäen globaaleja tyyppejä (global type)
  - paikallisessa määrittelyssä <xs:element> sisältää joko <xs:complexType> tai <xs:simpleType> elementin
  - globaalissa määrittelyssä viitataan valmiiksi määriteltyyn tyyppiin
    - voi olla joko Scheman tyyppi tai itse määritelty
    - itse määritellyt globaalit tyypit mahdollistavat dokumentin sirpaleiden (fragment) määrittelyn ja niiden käytön useassa eri kohdassa dokumenttia
    - globaali tyyppi määritellään tyypinmäärittelyelementillä <xs:complexType>, joka on <xs:schema> elementin lapsi ja siten viitattavissa koko schema-dokumentissa
    - Määritelty tyyppi nimetään name-attribuutilla, jotta siihen voidaan viitata

Globaalin tyypin määrittäminen

- Globaali tyyppi määritetään käyttäen elementtiä <complexType> schema-dokumentin juuritasolla (schema-elementin lapsielementtinä)

```
<schema>

   <complexType name="ChapterType">
    <sequence>
     <element name="para" type="string"/>
    </sequence>
   </complexType>
   …
</schema>
```

Globaali tyyppi

Globaalit elementit

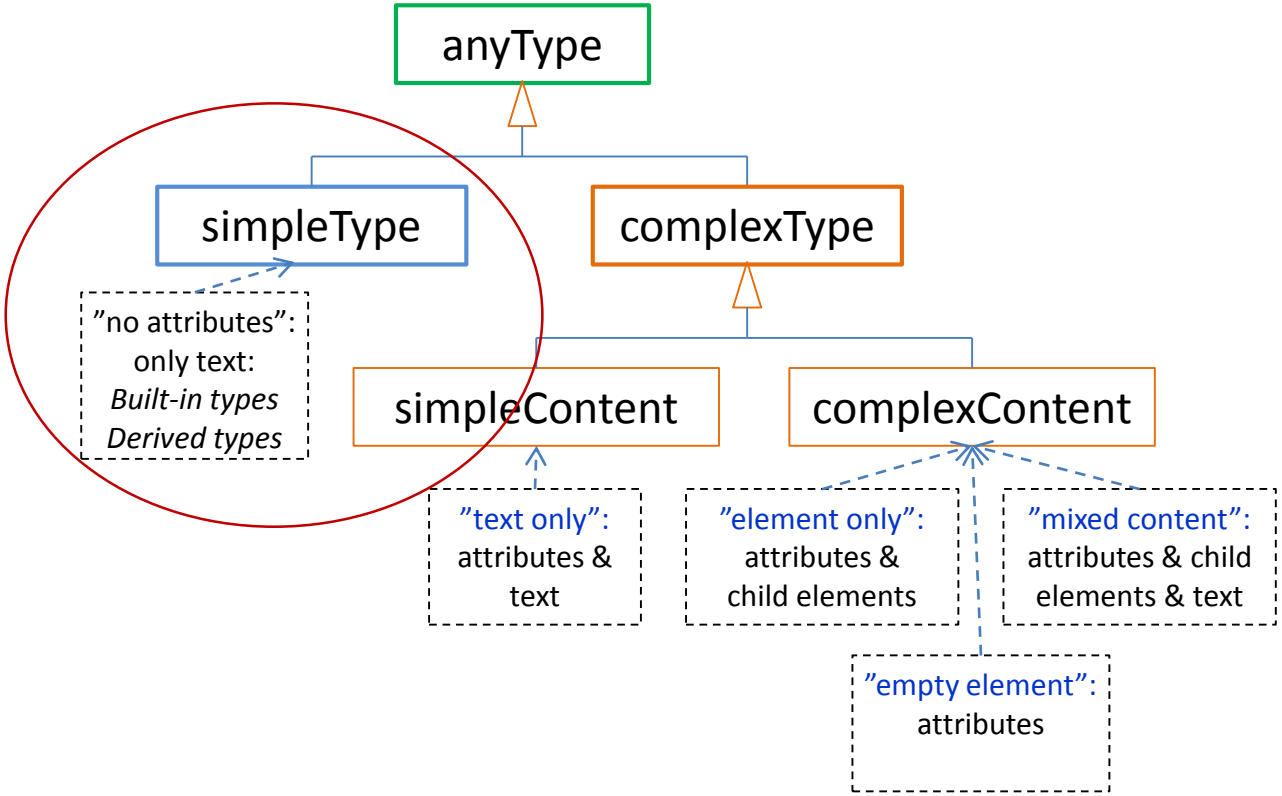
- Tyyppien määritysten* sijasta on mahdollista käyttää uudestaan jo määriteltyjä globaaleja elementtejä**
  - elementti määritellään normaalisti scheman juuritasolla
  - ko. elementtiin viitataan *ref*-attribuutilla määrittelyn sisältä:

```
<schema>
  <element name=”story" type="string"/>
  …
  <complexType name=”historyType">
   <sequence>
    <element ref=”story"/>
   </sequence>
  </complexType>
  …
  <element name=”history"  type=”won:historyType"/>
```

Globaali elementti

SIMPLE TYPE ELEMENTTI

Elementtien tyyppihierarkia



Elementti - Simple type

- Yksinkertaista tyyppiä (Simple type) oleva elementti voi sisältää
  - Vain tekstiä (numeroita, kirjaimia, erikoismerkkejä, unicode-merkkejä)
  - Mutta ei attribuutteja
- Tekstisisällön tyyppi voi olla esimerkiksi
  - *String*
  - *Integer*
  - *Boolean*
  - *Date*
- Tai mitä tahansa muuta XML-Scheman *built-in* tyyppiä (ks seur. Kalvo)
- Tai *built-in* tyypeistä johdettuja tyyppejä
- Tai yksinkertaisia tietorakenteita kuten
  - Range
  - Enumeration
  - List
  - Regex pattern

HUOM: Tyhjä elementti täytyy kuitenkin määritellä monitahoisena

Valmiit *built-in* tietotyypit

	Datatype
primitive	string
	boolean
	float
	double
	decimal
	duration
	dateTime
	time
	date
	gYearMonth
	gYear
	gMonthDay
	gDay
	gMonth
	hexBinary
	base64Binary
	anyURI
	QName
	NOTATION

derived	normalizedString
	token
	language
	IDREFS
	ENTITIES
	NMTOKEN
	NMTOKENS
	Name
	NCName
	ID
	IDREF
	ENTITY
	integer
	nonPositiveInteger
	negativeInteger
	long
	int
	short
	byte
	nonNegativeInteger
	unsignedLong
	unsignedInt
	unsignedShort
	unsignedByte
	positiveInteger

Built-in tietotyypit

- Xs:string
- Xs:integer
- Xs:date (YYYY-MM-DD)
- Xs.time (hh:mm:ss)
- Xs:dateTime (YYYY-MM-DD**T** hh:mm:ss)
- Duration (***P***n*Y*n*M*n*D**T**n*H*n*M*n*S*)

P=Period *T*=Time

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="simple_types">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="height" type="xs:string"/>
                <xs:element name="year_built" type="xs:integer"/>
                <xs:element name="birth" type="xs:date"/>
                <xs:element name="time_painted" type="xs:time"/>
                <xs:element name="when_shot" type="xs:dateTime"/>
                <xs:element name="strike_length" type="xs:duration"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

```
<?xml version="1.0"?>
<simple_types xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xsi:noNamespaceSchemaLocation="10-07.xsd">

    <height>39</height>
    <year_built>1280</year_built>
    <birth>1879-03-14</birth>
    <time_painted>21:08:00</time_painted>
    <when_shot>1968-04-04T18:01:00-05:00</when_shot>
    <strike_length>P5D</strike_length>

</simple_types>
```

-05:00= - 5 hour offset from UTC

Built-in tietotyypit

- Tekstikentän arvon etukäteismäärittely
  - Oletusarvo (default)
    - (the value of an element if it's empty or omitted)
  - Kiinteä arvo (fixed)
    - (the value of an element must be this value or the element omitted)

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="simple_types">
  <xs:complexType>
    <xs:sequence>
     <xs:element name="how_destroyed" type="xs:string" default="fire"/>
    </xs:sequence>
  </xs:complexType>
  </xs:element>
</xs:schema>
```

```
<?xml version="1.0"?>
<simple_types xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xsi:noNamespaceSchemaLocation="10-07.xsd">

    <how_destroyed>fire</how_destroyed>
    <how_destroyed></how_destroyed>
    <how_destroyed>earthquake</how_destroyed>

</simple_types>
```

Johdetut tietotyypit

- Johdetut tietotyypit
- <xs:restriction base=”xs:string ”>

```
<xs:element name="atomic_weight">
    <xs:simpleType>
        <xs:restriction base="xs:decimal">
            <xs:totalDigits value="6"/>
            <xs:fractionDigits value="4"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

```
<atomic_weight>12.0107</atomic_weight>
```

Johdetut tietotyypit

- Arvoalueen määrittely (range)

Lokaali määrittely:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="simple_types">
  <xs:complexType>

    <xs:element name="story">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:length value="1024"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>

    <xs:element name="total_bases">
      <xs:simpleType>
        <xs:restriction base="xs:integer">
          <xs:maxInclusive value="6856"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>

  </xs:complexType>
  </xs:element>
</xs:schema>
```

Globaali määrittely:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:simpleType name="story_type">
     <xs:restriction base="xs:string">
        <xs:length value="1024"/>
     </xs:restriction>
  </xs:simpleType>

  <xs:element name="simple_types">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="story" type="story_type"/>
            <xs:element name="summary" type="story_type"/>
            <xs:element name="another_story" type="story_type"/>
        </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Yksinkertaiset tietorakenteet

- Enumeration

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="simple_types">
    <xs:complexType>
      <xs:sequence>

        <xs:element name="wonder_name">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="Colossus of Rhodes"/>
                <xs:enumeration value="Great Pyramid of Giza"/>
                <xs:enumeration value="Hanging Gardens of Babylon"/>
                <xs:enumeration value="Statue of Zeus at Olympia"/>
                <xs:enumeration value="Temple of Artemis at Ephesus"/>
                <xs:enumeration value="Mausoleum at Halicarnassus"/>
                <xs:enumeration value="Lighthouse of Alexandria"/>
            </xs:restriction>
        </xs:simpleType>
        </xs:element>

      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Yksinkertaiset tietorakenteet

- List

```
...
<xs:element name="recent_eclipses">
    <xs:simpleType>
        <xs:list itemType="xs:dateTime"/>
    </xs:simpleType>
</xs:element>
...
```

```
<recent_eclipses>
    2010-12-21T08:17:00Z
    2011-06-15T20:13:00Z
    2011-12-10T14:32:00Z
</recent_eclipses>
```

```
...
<recent_eclipses>
    2008-02-21T03:26:00Z
    2007-08-28T10:37:00Z
</recent_eclipses>
...
```

Z=UTC

Yksinkertaiset tietorakenteet

- Regex pattern: Regular expressions = säännölliset lausekkeet

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="simple_types">
  <xs:complexType>
    <xs:sequence>

      <xs:element name="wonder_code">
          <xs:simpleType>
              <xs:restriction base="xs:string">
                  <xs:pattern value="w_\d{3}"/>
              </xs:restriction>
          </xs:simpleType>
      </xs:element>

    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

```
...
<wonder_code>w_285</wonder_code>
...
<race_time>PT2H4M26S</race_time>
...
```

```
...
<xs:element name="race_time">
    <xs:simpleType>
        <xs:restriction base="xs:duration">
            <xs:pattern value="PT\d+H\d+M\d+S"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
...
```

http://www.freeformatter.com/regex-tester.html

Yksinkertaiset tietorakenteet

- Regex ja union esimerkki:
- Regular expressions= säännölliset lausekkeet
- <xs:pattern> elementillä voidaan määritellä minkä muotoinen merkkijonon on oltava
- Sallittu muoto määritellään säännöllisellä lausekkeella
- Ks. Tarkemmin:
- Goldberg p. 132
- http://www.w3schools.com/jsref/jsref_obj_regexp.asp
- [http://fi.wikipedia.org/wiki/S%C3%A4%C3%A4nn%C3%B6llinen_lauseke]

```
<book>044508376X</book>

<book>978-0321559678</book>
```

```
<?xml version="1.0"?>
  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:simpleType name="isbn10">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{9}[\d|X]"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="isbn13">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{3}-\d{10}"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:element name="simple_types">
      <xs:complexType>
        <xs:sequence>

          <xs:element name="book">
            <xs:simpleType>
                <xs:union memberTypes="isbn10 isbn13"/>
            </xs:simpleType>
          </xs:element>

        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:schema>
```

RegExp – säännölliset lausekkeet

- A regular expression is an object that describes a pattern of characters.
- Regular expressions are used to perform pattern-matching and "search-and-replace" functions on text.

Brackets

Brackets are used to find a range of characters:

Expression	Description
[abc]	Find any character between the brackets
[^abc]	Find any character not between the brackets
[0-9]	Find any digit from 0 to 9
[A-Z]	Find any character from uppercase A to uppercase Z
[a-z]	Find any character from lowercase a to lowercase z
[A-z]	Find any character from uppercase A to lowercase z
[adgk]	Find any character in the given set
[^adgk]	Find any character outside the given set
(red|blue|green)	Find any of the alternatives specified

Quantifiers

Quantifier	Description
n+	Matches any string that contains at least one n
n*	Matches any string that contains zero or more occurrences of n
n?	Matches any string that contains zero or one occurrences of n
n{X}	Matches any string that contains a sequence of *X* n's
n{X,Y}	Matches any string that contains a sequence of X to Y n's
n{X,}	Matches any string that contains a sequence of at least X n's
n$	Matches any string with n at the end of it
^n	Matches any string with n at the beginning of it
?=n	Matches any string that is followed by a specific string n

Metacharacters

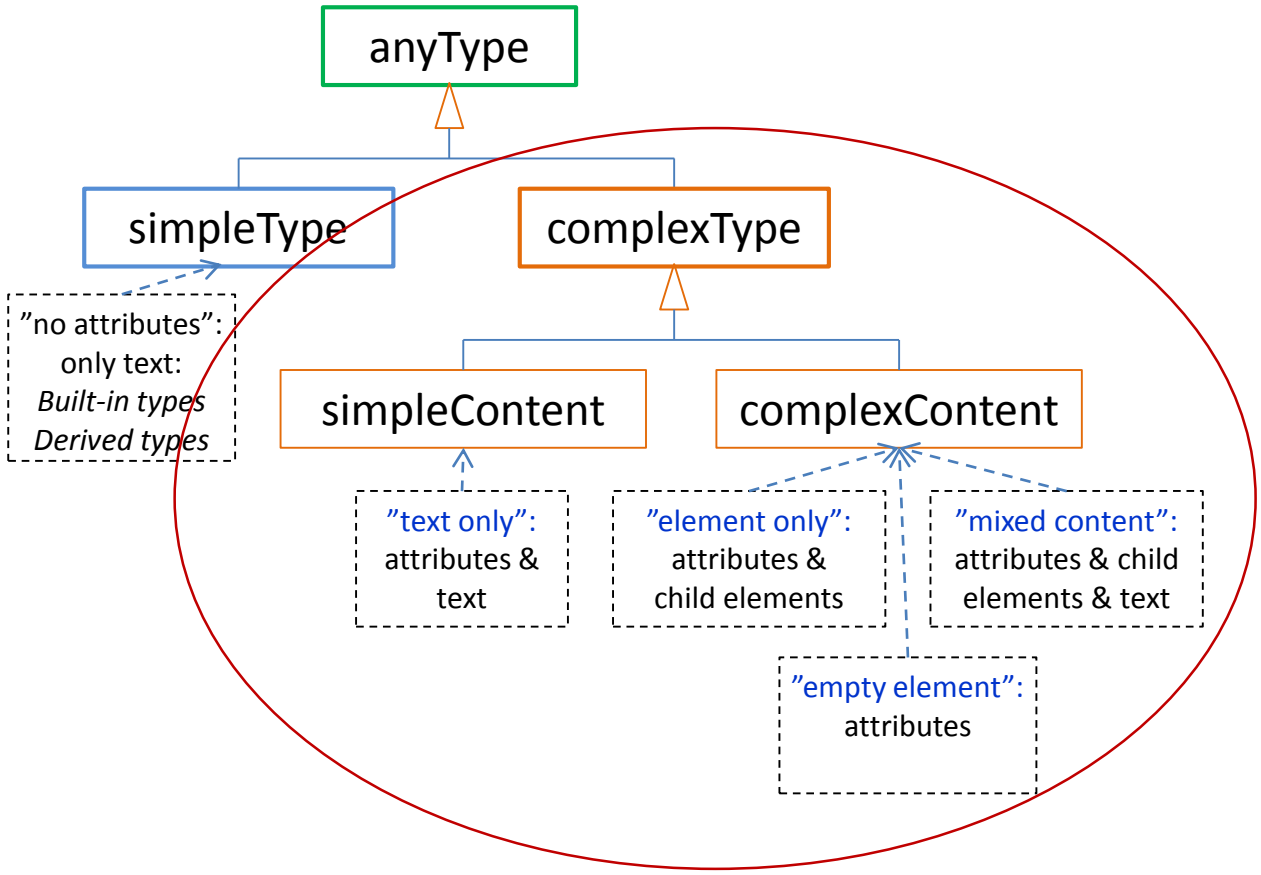
Metacharacters are characters with a special meaning:

Metacharacter	Description
.	Find a single character, except newline or line terminator
\w	Find a word character
\W	Find a non-word character
\d	Find a digit
\D	Find a non-digit character
\s	Find a whitespace character
\S	Find a non-whitespace character
\b	Find a match at the beginning/end of a word
\B	Find a match not at the beginning/end of a word
\0	Find a NUL character
\n	Find a new line character
\f	Find a form feed character
\r	Find a carriage return character
\t	Find a tab character
\v	Find a vertical tab character
\xxx	Find the character specified by an octal number xxx
\xdd	Find the character specified by a hexadecimal number dd
\uxxxx	Find the Unicode character specified by a hexadecimal number xxxx

[http://www.w3schools.com/jsref/jsref_obj_regexp.asp]
http://www.freeformatter.com/regex-tester.html

COMPLEX TYPE ELEMENTTI

Elementtien tyyppihierarkia



Monitahoinen elementti

- Monitahoinen (complex type)
  - lapsielementtejä ja/tai attribuutteja sisältävät elementit
  - Ei ole valmiiksi määriteltyjä tyyppejä
  - Kaikki monitahoiset tyypit täytyy johtaa
- Monitahoiset elementit jaetaan edelleen kahteen alaryhmään
  - Yksinkertainen tietosisältö (simple content)
    - Täytyy eksplisiittisesti määritellä
  - Monitahoinen tietosisältö (complex content)
    - HUOM: Oletussisältötyyppi, joten tätä alaryhmää ei tarvitse aina määritellä

HUOM: Oletus on monitahoinen sisältö tyyppi (complex type::complex content), joka johdetaan anyType:stä [Goldberg p. 139]

Simple content – ”text only”

- Yksinkertainen tietosisältö (simple content)
  - Attribuutteja & tekstiä
- Täytyy eksplisiittisesti määritellä elementillä:
  <xs:simpleContent>

```
...
<xs:element name="year_built">
  <xs:complexType>
    <xs:simpleContent>
        <xs:extension base="xs:positiveInteger">
            <xs:attribute name="era" type="xs:string"/>
        </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
...
```

HUOM: Oletus on monitahoinen sisältö tyyppi (complex type::complex content), joka johdetaan anyType:stä [Goldberg p. 139]

Complex content

```
<xs:element name="ancient_wonders">
  <xs:complexType>
    <xs:complexContent>
        <xs:restriction base="xs:anyType">
            <xs:sequence>
                <xs:element name="wonder" type="wonde
            </xs:sequence>
        </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
```

Jätetty pois

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="ancient_wonders">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="wonder" type="wonderType"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="wonderType">
        <xs:sequence>
            <xs:element name="name" type="nameType"/>
            <xs:element name="location" type="xs:string"/>
            <xs:element name="height" type="heightType"/>
            <xs:element name="history" type="historyType"/>
            <xs:element name="main_image" type="imageType"/>
            <xs:element name="source" type="sourceType"/>
        </xs:sequence>
    </xs:complexType>
    ...
```

- Oletustyyppi: *xs:anyType* tyypistä rajoittamalla johdettu *xs:complexContent* tietotyyppi
- Koska se on oletusarvo se usein jätetään merkitsemättä (mikä on hiukan hämäävää)
- Johdettujen tyyppien tapauksessa *xs:complexContent*-elementti täytyy kuitenkin esittää (ks. Kalvo 45)
- Johdettu *xs:simpleContent* tietotyyppi täytyy aina eksplisiittisesti määritellä

```
...
<xs:element name="year_built">
  <xs:complexType>
    <xs:simpleContent>
        <xs:extension base="xs:positiveInteger">
            <xs:attribute name="era" type="xs:string"/>
        </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
...
```

Esimerkki: Simple & complex content

```
<!-- Named complex type defined globally -->
<xs:complexType name="yearType">
    <xs:simpleContent>
        <xs:extension base="xs:positiveInteger">
            <xs:attribute name="era" type="xs:string"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>



<xs:complexType name="historyType">
    <xs:sequence>
        <xs:element name="year_built" type="yearType"/>
        <xs:element name="year_destroyed" type="yearType"/
        <xs:element name="how_destroyed" type="destrType"/
        <xs:element name="story" type="storyType"/>
    </xs:sequence>
</xs:complexType>
```

```
<?xml version="1.0"?>
<ancient_wonders xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:noNamespaceSchemaLocation="11-10.xsd">
    <wonder>
        <name language="English">Colossus of Rhodes</name>
        <location>Rhodes, Greece</location>
        <height units="feet">107</height>
        <history>
            <year_built era="BC">282</year_built>
            <year_destroyed era="BC">226</year_destroyed>
            <how_destroyed>earthquake</how_destroyed>
            <story>
            In 294 BC, the people of the island of
            Rhodes began building a colossal statue
            of the sun god Helios.
            </story>
        </history>
        <main_image file="colossus.jpg" w="528" h="349"/>
        <source sectionid="101" newspaperid="21"/>
    </wonder>
</ancient_wonders>
```

Huom:
oletusmäärittely on jätetty pois:
<xs:complexContent>
<xs:restriction base="xs:anyType”>

Complex type::Complex content elementit

SISÄLTÖMALLIT

Content models - sisältömallit

- Elementtejä sisältävän monitahoisen tyypin määrittely
- Sisältömalli (content model):
  - Määriteltävän tyypin lapsielementit
  - Rakenteen ja lapsielementtien järjestyksen määrittelevä malliryhmä (model group)
    1. Elementtien järjestetty sekvenssi (sequence)
    2. Elementtien järjestämätön lista (all)
    3. Elementtien vaihtoehtoiset valinnat (choise)

Model group

```
...
<xs:complexType name="wonderType">
    <xs:sequence>
        <xs:element name="name" type="nameType"/>
        <xs:element name="location" type="xs:string"/>
        <xs:element name="height" type="heightType"/>
        <xs:element name="history" type="historyType"/>
        <xs:element name="main_image" type="imageType"/>
        <xs:element name="source" type="sourceType"/>
    </xs:sequence>
</xs:complexType>
...
```

Huom:
oletussisältömäärittely on jätetty pois:
<xs:complexContent>
<xs:restriction base="xs:anyType”>

Sisältömalli - sekvenssi

- *Sequence*-elementti määrittelee tyypin lapsielementtien järjestetyn listan
  - Sekvenssin esiintymiskerrat määritellään attribuuteilla *minOccurs* ja *maxOccurs*
  - Tietotyypin sekvenssi-malliryhmään kuuluvat lapsielementit määritellään *xs:element*-elementillä
    - *name*-attribuutin arvo määrää lapsielementin nimen
    - lapsielementin mahdollinen lukumäärä määritellään *minOccurs* ja *maxOccurs* attribuuteilla (molempien oletusarvo on yksi 1)
  - Sekvenssissä määriteltyjen lapsielementtien tulee esiintyä validissa XML-instanssidokumentissa sekvenssi-määrittelyn mukaisessa järjestyksessä
  - Sequence-malliryhmä voi sisältää toisia sequence- ja/tai choise-malliryhmiä

Sisältömalli – järjestämätön lista

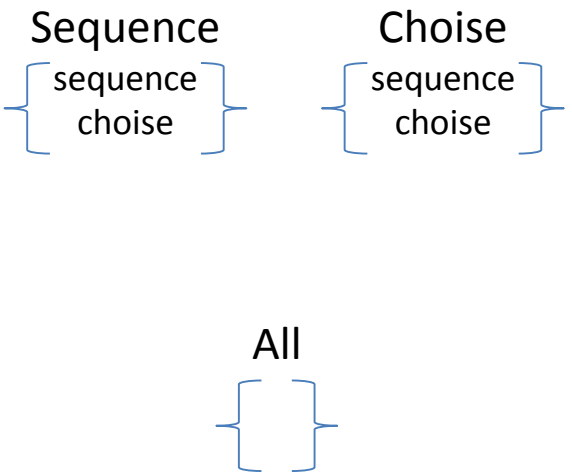
- *All*-elementti määrittelee tyypin lapsielementtien järjestämättömän listan
  - All-malliryhmän esiintymiskerrat määritellään attribuuteilla *minOccurs* ja *maxOccurs*, mutta niiden arvot voivat olla vain nolla (0) tai yksi (1)
  - Tietotyypin all-malliryhmään kuuluvat lapsielementit määritellään *xs:element*-elementillä
    - *name*-attribuutin arvo määrää lapsielementin nimen
    - Lapsielementin mahdollinen lukumäärä määritellään *minOccurs* ja *maxOccurs* attribuuteilla, jotka voivat saada vain arvot nolla (0) tai yksi (1)
  - All-malliryhmässä määritellyt lapsielementit voivat esiintyä validissa XML-instanssidokumentissa missä tahansa järjestyksessä
  - Jos tietotyyppi määrittelee All-malliryhmän, muita malliryhmiä ei voida määritellä

Sisältömalli - vaihtoehdot

- *Choise*-elementti määrittelee tyypin lapsielementtien mahdolliset vaihtoehtoiset lapsielementtivalinnat
  - Choice-malliryhmän esiintymiskerrat määritellään attribuuteilla *minOccurs* ja *maxOccurs*
  - Tietotyypin Choice-malliryhmään kuuluvat lapsielementit määritellään *xs:element*-elementillä
    - name-attribuutin arvo määrää lapsielementin nimen
    - lapsielementin mahdollinen lukumäärä määritellään *minOccurs* ja *maxOccurs* attribuuteilla (molempien oletusarvo on yksi 1)
  - Vain yksi Choice-malliryhmän lapsielementeistä saa esiintyä validissa XML-instanssidokumentissa (lukumäärän tulee olla kardinaliteettimäärittelyjen mukainen)
  - Choise-malliryhmä voi sisältää sequence- ja/tai choise-malliryhmiä

Malliryhmien sisäkkäisyys

- Malliryhmät sequence ja choise saavat sisältää toisiaan
- Mutta all-malliryhmä ei saa sisältää muita malliryhmiä ja sen tulee olla ainoa tyyppimäärittelyn malliryhmä



Esimerkki - malliryhmät

Sequence model group

```
...
<xs:complexType name="wonderType">
    <xs:sequence>
        <xs:element name="name" type="nameType"/>
        <xs:element name="location" type="xs:string"/>
        <xs:element name="height" type="heightType"/>
        <xs:element name="history" type="historyType"/
        <xs:element name="main_image" type="imageType"
        <xs:element name="source" type="sourceType"/>
    </xs:sequence>
</xs:complexType>
...
```

Nested Sequence and Choise model groups

```
...
<xs:complexType name="wonderType">
    <xs:sequence>
        <xs:element name="name" type="nameType"/>

        <xs:choice>
            <xs:element name="location" type="xs:string"/>
            <xs:sequence>
                <xs:element name="city" type="xs:string"/>
                <xs:element name="country" type="xs:string"
            </xs:sequence>
        </xs:choice>

        <xs:element name="height" type="heightType"/>
        <xs:element name="history" type="historyType"/>
        <xs:element name="main_image" type="imageType"/>
        <xs:element name="source" type="sourceType"/>
    </xs:sequence>
</xs:complexType>
...
```

Esimerkki - malliryhmät

All model group

```
<xs:complexType name="historyType">
    <xs:all>
        <xs:element name="year_built" type="yearType"/>
        <xs:element name="year_destroyed" type="yearType"/>
        <xs:element name="how_destroyed" type="destrType"/>
        <xs:element name="story" type="storyType"/>
    </xs:all>
</xs:complexType>
```

```
<?xml version="1.0"?>
<ancient_wonders xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:noNamespaceSchemaLocation="11-10.xsd">
    <wonder>
        <name language="English">Colossus of Rhodes</name>
        <location>Rhodes, Greece</location>
        <height units="feet">107</height>
        <history>
            <year_built era="BC">282</year_built>
            <year_destroyed era="BC">226</year_destroyed>
            <how_destroyed>earthquake</how_destroyed>
            <story>
            In 294 BC, the people of the island of
            Rhodes began building a colossal statue
            of the sun god Helios.
            </story>
        </history>
        <main_image file="colossus.jpg" w="528" h="349"/>
        <source sectionid="101" newspaperid="21"/>
    </wonder>
</ancient_wonders>
```

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

<ancient_wonders xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:noNamespaceSchemaLocation="11-16.xsd">
    <wonder>
        <name language="English">Colossus of Rhodes</name>
        <location>Rhodes, Greece</location>
        <height units="feet">107</height>
        <history>
            <story>
        In 294 BC, the people of the island of Rhodes
        began building a colossal statue of the sun god Helios.
            </story>
            <year_built era="BC">282</year_built>
            <how_destroyed>earthquake</how_destroyed>
            <year_destroyed era="BC">226</year_destroyed>
        </history>
        <main_image file="colossus.jpg" w="528" h="349"/>
        <source sectionid="101" newspaperid="21"/>
    </wonder>
</ancient_wonders>
```

”mixed content” & ”empty element”

- Ks. Goldberg sivu 148 & 147

- Ks. http://www.w3.org/XML/Schema#dev

Online validation tool: http://www.xpathtester.com/validate

Johdetut monitahoiset tyypit

- Jo määritellyistä monitahoisista tyypeistä voidaan edelleen johtaa uusia monitahoisia tyyppejä
- Johdetulla tyypillä on aluksi perustyypin (base type) kaikki ominaisuudet; johtamisessa ominaisuuksien joukkoa joko
  - laajennetaan (extension) tai
  - rajoitetaan (restriction)
- Määrittelyn rakenne:

  <xs:complexType name="uusi nimi">
  <xs:complexContent>
  <xs:extension base="perustyyppi"> tai
  <xs:restriction base="perustyyppi">

```
<xs:complexType name="historyType">
    <xs:sequence>
        <xs:element name="year_built" type="yearType"/>
        <xs:element name="year_destroyed" type="yearType"/>
        <xs:element name="how_destroyed" type="destrType"/>
        <xs:element name="story" type="storyType"/>
    </xs:sequence>
</xs:complexType>
```

Base type: historyType

extension

```
<xs:complexType name="newHistoryType">
  <xs:complexContent>
    <xs:extension base="historyType">
        <xs:sequence>
            <xs:element name="who_built" type="xs:string"/>
        </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Derived type: newHistoryType

lisätään uusi elementti

restriction

```
<xs:complexType name="newHistoryType">
  <xs:complexContent>
    <xs:restriction base="historyType">
      <xs:sequence>
        <xs:element name="year_built" type="yearType"/>
        <xs:element name="year_destroyed" type="yearType"/>
        <xs:element name="how_destroyed" type="destrType" fixed="fire"/>
        <xs:element name="story" type="storyType"/>
      </xs:sequence>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
```

Derived type: newHistoryType

kiinnitetään elementin arvo

HUOM: Määrittelyssä on nyt käytettävä <xs:complexContent> elementtiä

ATTRIBUUTTIEN MÄÄRITTELY

Attribuuttien määrittely

- Attribuuttien määrittely
- Kaikki elementit, joilla on attribuutteja kuluvat, johonkin neljästä monitahoisen tyypin (complex type) ryhmästä
- Attribuutit itsessään ovat aina yksitahoista tyyppiä (simple type)
- Attribuutit määritellään elementillä *<xs:attribute>*
  - Sen *name*-attribuutin arvo määrää attribuutin nimen
  - attribuutin tyyppi määritellään joko *type*-attribuutin arvolla tai rajoittamalla (restriction) tai laajentamalla (extension) jotain *base*-tyyppiä

Attribuutit

```
...
<xs:complexType name="sourceType">
    <xs:attribute name="sectionid"
        type="xs:positiveInteger"
        use="required"/>
    <xs:attribute name="newspaperid">
        <xs:simpleType>
            <xs:restriction base="xs:positiveInteger">
                <xs:pattern value="\d{4}"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>
...
```

restriction

```
...
<source sectionid="141" newspaperid="9999"/>
<source sectionid="2"/>
...
```

```
<xs:complexType name="videoType">
    <xs:attributeGroup ref="imageAttrs"/>
    <xs:attribute name="format" type="xs:string"/>
</xs:complexType>

<xs:attributeGroup name="imageAttrs">
    <xs:attribute name="file" type="xs:anyURI" use="required"/>
    <xs:attribute name="w" type="xs:positiveInteger" use="required"/>
    <xs:attribute name="h" type="xs:positiveInteger" use="required"/>
</xs:attributeGroup>
```

Esimerkki: XML -scheman vaiheittainen määrittely xml dokumentille

XML-SCHEMAN VAIHEITTAINEN MÄÄRITTELY

Esimerkki: XML-Scheman vaiheittainen kehitys

- Tavoitteena kirjoittaa xml-schema xml-dokumentille:
  - *unit_cells_example.xml*
- Edetään vaiheittain:
  1. Aluksi sallitaan kaikenlaiset rakenteet
  2. Vaiheittain määritellään elementtien tyypit ja niiden sallitut lapsi-elementit
  3. Ei rajoiteta elementtien attribuutteja
  4. Kun elementti-rakenne on määritelty, aloitetaan attribuuttien määrittely
  5. Lopuksi määritellään halutut elementtien lukumäärät ja elementtien ja attribuuttien tarkat tyypit
- Kaikissa vaiheissa kannattaa tarkistaa, että kohdedokumentti on validi xml-scheman suhteen.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- XML document: unit_cells_example.xml
<unit_line>
  <unit>
    <assembly_cell>
      <name></name>
      <welding></welding>
      <sensor></sensor>
      <sensor></sensor>
    </assembly_cell>
  </unit>

  <unit>
    <packaging_cell>
      <name></name>
      <conveyor></conveyor>
      <conveyor></conveyor>
      <sensor></sensor>
      <sensor></sensor>
    </packaging_cell>
  </unit>

</unit_line>
```

[Demo with : http://www.xpathtester.com/validate]

XML-Scheman kehitys: Vaihe 0

- Aloitetaan schemasta, joka rajoittaa mahdollisimman vähän validoitavan dokumentin rakennetta (eli sallii lähes kaiken)
- *unit_cells_xsd_0.xsd*
  - Nimetään vain juurielementti globaalilla elementti-määrittelyllä
  - Elementin pelkkä nimeäminen ei rajoita sen tyyppiä vielä lainkaan
- Kohdedokumentti *unit_cells_example.xml* on validi tämän scheman suhteen

```
<?xml version="1.0" encoding="utf-8"?>
<!-- unit_cells_xsd0.xsd -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
<!-- Global element definitions -->
<xs:element name="unit_line"/>
</xs:schema>
```

Juurielementti

[Demo with : http://www.xpathtester.com/validate]

XML-Scheman kehitys: Vaihe 1

- Seuraavaksi rajoitetaan hiukan lisää validoitavan dokumentin rakennetta
- *unit_cells_xsd1.xsd*
  - Nimetään juurielementin lapset globaalilla elementti-määrittelyllä
  - Määritellään juurielementin tyyppi: *xs:complexType*
    - Sisältää lapsielementtejä joihin viitataan *ref-attribuutilla*
    - Ei rajoitetta lapsien lukumäärää *minOccurs=0 ja maxOccurs=unbounded*

- Kohdedokumentti *unit_cells_example.xml* on validi tämänkin scheman suhteen

```
<?xml version="1.0" encoding="utf-8"?>
<!-- unit_cells_xsd1.xsd -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault
    <!-- Global element definitions -->
    <xs:element name="unit"/>

    <xs:element name="unit_line">
        <xs:complexType>
            <xs:sequence>
            <xs:element ref="unit" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Juurielementin lapsi

Juurielementin tyyppimäärittely

XML-Scheman kehitys: Vaihe 2

- Seuraavaksi rajoitetaan hiukan lisää validoitavan dokumentin rakennetta
- *unit_cells_xsd2.xsd*
  - Nimetään juurielementin yhden lapsen lapsielementit globaalilla elementti-määrittelyllä
  - Määritellään juurielementin yhden lapsen tyyppi: *xs:complexType*
    - Sisältää lapsielementtejä joihin viitataan *ref-attribuutilla*
    - Ei rajoitetta lapsien lukumäärää *minOccurs=0 ja maxOccurs=unbounded*
  - Ei rajoiteta attribuutteja tyyppimäärittelyssä lisäämällä määrittelyyn elementti:
    - <xs:anyAttribute processContents="skip"/>

- Kohdedokumentti *unit_cells_example.xml* on validi tämänkin scheman suhteen
- Näin edetään vaiheittain kunnes elementti-rakenne on määritelty
- Tämän jälkeen määritellään elementeiltä vaaditut attribuutit.

Kuva: unit_cells_xsd2.xsd seuraavalla kalvolla

XML-Scheman kehitys: Vaihe 2

- unit_cells_xsd2.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<!-- unit_cells_xsd2.xsd -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
<!-- Global element definitions -->
<!-- unit's child element definitions -->
<xs:element name="assembly_cell"/>
<xs:element name="packaging_cell"/>

    <xs:complexType name="unitType">
        <xs:choice>
            <xs:element ref="assembly_cell"/>
            <xs:element ref="packaging_cell"/>
        </xs:choice>
        <!-- Note: no attribute constraints -->
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:element name="unit_line">
        <xs:complexType>
            <xs:sequence>
            <xs:element name="unit" type="unitType" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

Unit-elementin lapset

Lapsielementin tyyppimäärittely

XML-Scheman kehitys: Vaihe 3

- unit_cells_xsd3.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<!-- unit_cells_xsd2.xsd -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
<!-- Global element definitions -->
<!-- unit's child element definitions -->
<!--xs:element name="assembly_cell"/-->
<xs:element name="packaging_cell"/>
<!-- assembly_cell's child element definitions -->
<xs:element name="name"/>
<xs:element name="welding"/>
<xs:element name="sensor"/>

    <xs:complexType name="unitType">
        <xs:choice>
            <xs:element name="assembly_cell" type="assembly_cell_type"/>
            <xs:element ref="packaging_cell"/>
        </xs:choice>
        <!-- Note: no attribute constraints -->
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:element name="unit_line">
        <xs:complexType>
            <xs:sequence>
            <xs:element name="unit" type="unitType" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="assembly_cell_type">
        <xs:sequence>
            <xs:element ref="name" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="welding" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element ref="sensor" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <!-- Note: no attribute constraints -->
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

</xs:schema>
```

Lähdeviitteet

- Kurssikirja:
  - XML-Schemaan liittyvät luvut 9, 10, 11 ja 13 kirjasta: Goldberg, K.,H.2009. XML-Visual QuickStart Guide. 2.e. Peachpit Press XML. : Introduction, Ch1, Ch2, Ch3, Ch4 ja Ch9-11, Ch12, Ch13, Ch14-15.
  - Goldberg kirjan esimerkit: http://www.kehogo.com/examples
  - XML-Schemaan liittyvät luku 5 ja Appendix C kirjasta: E-book: Beginning XML, (5th ed.) by Fawcett, J., et al., John Wiley & Sons, Inc. Ch 1-2, Ch3, **Ch5**, Ch7, Ch8.
- Muita lähteitä:
  - Tutorials: http://www.w3schools.com/
  - W3C:n Määrittelyt:
  - http://www.w3.org/XML/Core/
  - http://www.w3.org/Style/XSL/;
  - http://www.w3.org/XML/Schema#dev
- Tools
  - Online tool: xpath, xslt, schema validate: http://www.xpathtester.com
  - Online tool: Regular Expression Tester: http://www.freeformatter.com/regex-tester.html
  - Command line tool: Xmlstarlet command line XML toolkit:
    - Download for windows: http://xmlstar.sourceforge.net/download.php