Semantic Web

Tutorial 3-4

Iryna Dubrovska

WeST
People and Knowledge Networks

Tutorial 3

Task 1. XML and XML Schema

Main rules to reach well-formed XML:

- XML must have a root element;
- All start tags must have closing tags;
- Elements are case sensitive (!);
- Elements must be properly nested;
- Attribute values must be quoted;
- An element cannot have two attributes with the same name;
- Be careful with characters (<, &, etc). Use entity references.

Well-formed is not enough!

Valid XML:

- Well-formed;
- Conforms to its schema (DTD or XML Schema)

https://www.w3schools.com/xml/xml_elements.asp

Task 1. XML and XML Schema

1. Check the validity of XML according to its schema. Point out the issues and rewrite XML accordingly.

```
1  <!DOCTYPE record [
2    <!ELEMENT record (artist |year |contributor)+>
3    <!ELEMENT artist (record)+>
4    <!ELEMENT year (#PCDATA)>
5    <!ELEMENT contributor (#PCDATA)>
6  ]>
7
8  <record>
9    <artist>
10     <record>text</record>
11   </artist>
12   <year>text</year>
13   <contributor>text</contributor>
14 </record>
```

Answer:

1. Rule <!ELEMENT artist (record)+> says that there has to be at least one record inside of *artist*. There is **no rule** to specify that the element *record* could be char data type. Therefore, writing
<artist><record>text</record></artist>
is inappropriate.

2. An element *record* requires a child element, which is not provided. Therefore, writing
<artist><record>text</record></artist> or
<artist><record></record></artist>
is inappropriate.

3. It would be nice to specify the **encoding** of the document
<?xml version="1.0" encoding="UTF-8"?>

```
18 <?xml version="1.0" encoding="UTF-8"?>
19 <!DOCTYPE record [
20   <!ELEMENT record ((artist |year |contributor)+)>
21   <!ELEMENT artist (record)+>
22   <!ELEMENT year (#PCDATA)>
23   <!ELEMENT contributor (#PCDATA)>
24 ]>
25 <record>
26   <artist>
27     <record><year/></record>
28   </artist>
29   <year>text</year>
30   <contributor>text</contributor>
31 </record>
```

Task 1. XML and XML Schema

2. Translate the above DTD into an XML Schema.

Building blocks of XML Schema:

- Elements from the namespace http://www.w3.org/2001/XMLSchema
- Attributes from the namespace http://www.w3.org/2001/XMLSchema

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE record [
  <!ELEMENT record ((artist |year |contributor)+)>
  <!ELEMENT artist (record)+>
  <!ELEMENT year (#PCDATA)>
  <!ELEMENT contributor (#PCDATA)>
]>
```

```
1  <?xml version="1.0"?>
2  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
3    <xs:element name="record" type="TypeRecord"/>
4    <xs:element name="artist">
5      <xs:complexType>
6        <xs:sequence>
7          <xs:element ref="record" minOccurs="1" maxOccurs="unbounded"/>
8        </xs:sequence>
9      </xs:complexType>
10   </xs:element>
11   <xs:element name="year" type="xs:string"/>
12   <xs:element name="contributor" type="xs:string"/>
13   <xs:complexType name="TypeRecord">
14     <xs:choice minOccurs="1" maxOccurs="unbounded">
15       <xs:element ref="artist"/>
16       <xs:element ref="year"/>
17       <xs:element ref="contributor"/>
18     </xs:choice>
19   </xs:complexType>
20 </xs:schema>
```

https://www.w3schools.com/xml/xml_elements.asp

Task 2. Python Programming

```python
from lxml import etree
import xml.etree.ElementTree as ET

XMLFile = "valid_xml.xml"
schemaFile = "schema.xsd"
DTDFile = "schema.dtd"

#Task 2a
def validateSchema(XMLFile, schemaFile):
    schemaDoc = etree.parse(schemaFile)
    xmlSchema = etree.XMLSchema(schemaDoc)
    xml = etree.parse (XMLFile)
    if xmlSchema.validate(xml):
        print("File Validated")
    else:
        print("XML file does not conform with its Schema")
    for error in xmlSchema.error_log:
        print(error)

def validateDTD (XMLFile, DTDFile):
    xml_validator = etree.DTD(DTDFile)
    f = etree.parse(XMLFile)
    if xml_validator.validate(f):
        print("File Validated")
    else :
        print("XML file does not conform with its DTD")
    for error in xml_validator.error_log :
        print(error)
```

```python
#Task 2b
def validateXMLFile (XMLFile,schemaFile):
    exts = schemaFile.split('.')
    if(exts[1] == 'dtd'):
        validateDTD(XMLFile,DTDFile)
    elif exts [1] == 'xsd':
        validateSchema(XMLFile,schemaFile)
    else :
        print('Not a valid extension')
        print(exts[1])

#Task 2c
tree = ET.parse('valid_xml.xml')
root = tree.getroot()

for contributor in root.iter('contributor'):
    contributor.text = str(contributor.text) + " Irina"
    contributor.set('gender', 'female')
    mytree.write('result.xml')

#Task 2d
new_tree = ET.parse('result.xml')
new_root = new_tree.getroot()

for contributor in new_root.iter('contributor'):
    contributor.attrib.pop('gender', None)
new_tree.write('result.xml')
```

Tutorial 4

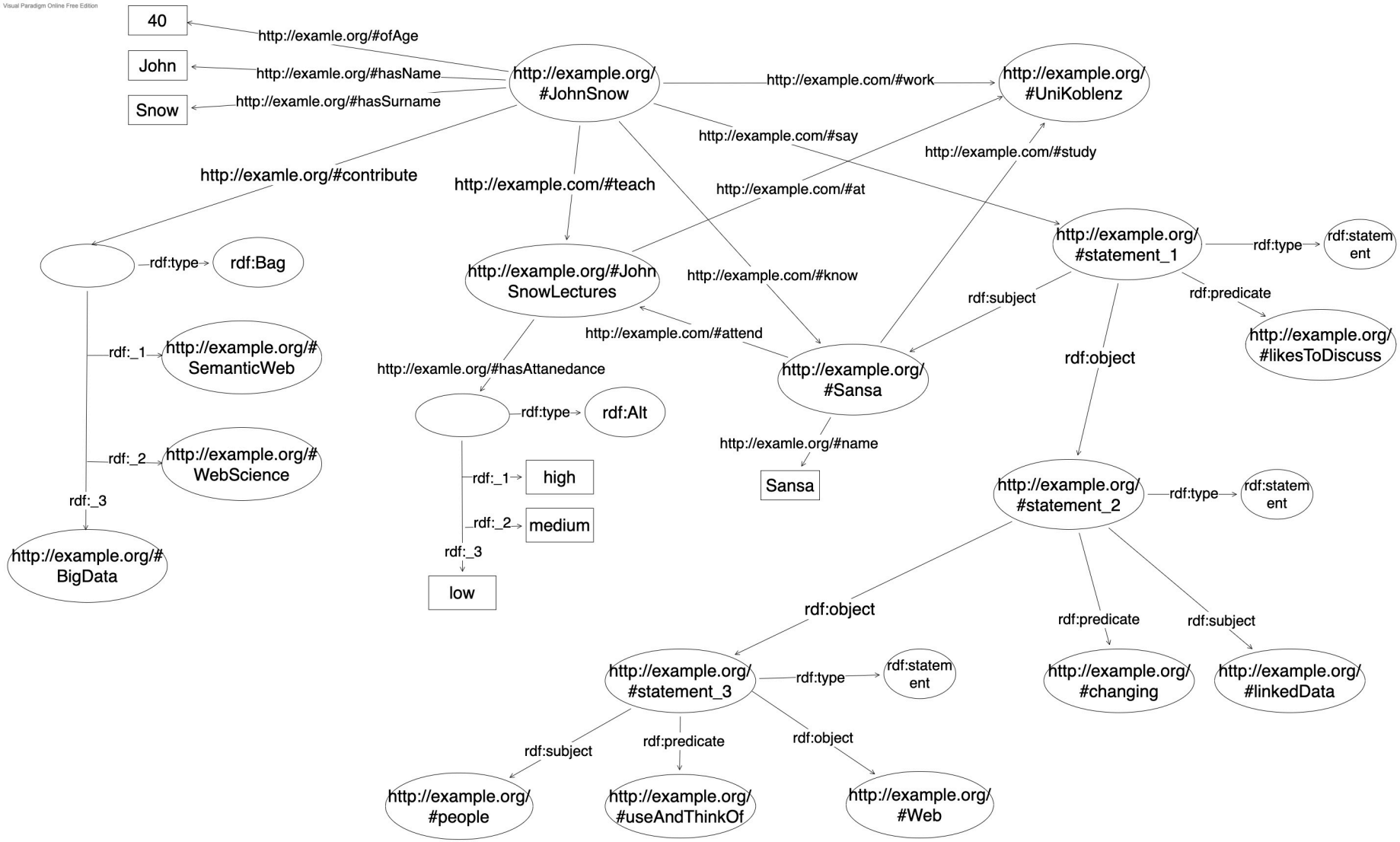
WeST
People and Knowledge Networks

Task 1. Managing RDF

1. John Snow (*http://example.org/#John_Snow)* is 40 years old and he gives lectures (*http: //example.org/#John_Snow_lectures*) at the university of Koblenz-Landau (*http:// example.org/#Uni_Koblenz_Landau*). He also contributes to the courses Semantic Web (*http://example.org/#Semantic_Web*), Web Science (*http://example.org/#Web_Science*), and Big Data (*http://example.org/#Big_Data*). He has a friend Sansa Stark (*http: //example.org/#Sansa_Stark)* who studies at the same university, and participates in his lectures. His lectures actually have a pretty high attendance. He says (http://example. org/#says) that Sansa likes to discuss (http://example.org/#likes_toDiscuss) how linked data is changing the way people use and think of (http://example.org/#use_ thinkOf) the Web.

   Translate the above scenario into a visual RDF graph representation using the graphical notation presented in the lecture. Some hints regarding the representation are already given to you in the task. Pay attention to the tricky part written in italic.

Task 1. Managing RDF



Task 1. Managing RDF

2. Translate the graph into RDF/XML document. Make sure your XML is valid.

```
<?xml version = "1.0" encoding = "utf-8" ?>
<rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:xo = "http://example.org/ontology#">
  <rdf:Description rdf:about = "http://example.org/resource/Vancouver" >
    <rdf:type rdf:resource = "http://example.org/ontology#City" />
    <xo:name> Vanvouver </xo:name >
    <xo:latitude> 49.25 </xo:latitude >
    <xo:longitude> -123.1 </xo:longitude >
    <xo:population> 631,000 </xo:population >
    <xo:state rdf:resource = "http://example.org/resource/British_Columbia" />
    <xo:mayor >
      <xo:Person rdf:about = "http://example.org/resource/Gregor_Robertson">
        <xo:age> 53 </xo:age >
        <xo:assumedOffice> 2008-12-08 </xo:assumedOffice >
        <xo:party rdf:resource = "http://example.org/resource/BC_NDP" />
      </xo:Person>
    </xo:mayor>
  </rdf:Description>
</rdf:RDF>
```

Task 2. Python Programming

```
import rdflib
from rdflib import URIRef
```

```
g = rdflib.Graph()
```

```
format = rdflib.util.guess_format("http://dbpedia.org/data/Berlin.rdf")
g.parse("http://dbpedia.org/data/Berlin.rdf", format=format)
```

```
<Graph identifier=Na0af9ed5e49341e59e0c3c0e6459ede6 (<class 'rdflib.graph.Graph'>)>
```

```
def relatedTo(propertyURI):
    print("Related to property",propertyURI, ":")
    for resource in g.subjects(URIRef(propertyURI)):
        print(resource)
```

```
URI = "http://dbpedia.org/property/birthPlace"
relatedTo(URI)
```

```
Related to property http://dbpedia.org/property/birthPlace :
http://dbpedia.org/resource/Björn_Andrae
http://dbpedia.org/resource/Carl_Linger
http://dbpedia.org/resource/Hermann_Knoblauch
http://dbpedia.org/resource/Hermann_von_Oppeln-Bronikowski
http://dbpedia.org/resource/Kurt_von_Tippelskirch
http://dbpedia.org/resource/Sebastian_Stefaniszin
http://dbpedia.org/resource/Volker_Berghahn
http://dbpedia.org/resource/Wilhelm_Heinrich_Heintz
http://dbpedia.org/resource/Albert_Wodrig
http://dbpedia.org/resource/Carsten_Bresch
http://dbpedia.org/resource/Charlotte_Bischoff
http://dbpedia.org/resource/Erich_Fuchs
http://dbpedia.org/resource/Ernst-Anton_von_Krosigk
http://dbpedia.org/resource/Ernst_Heilmann
http://dbpedia.org/resource/Erwin_Thiesies
```

Questions?