Biomedical semantics, information retrieval and knowledge discovery - (6)

Uniform Resource Identifiers (URI)

Dr. Dietrich Rebholz-Schuhmann
Dr. Leyla Jael Garcia Castro

December 18, 2020

Outline

Objectives and learning outcomes

Objectives

- We will introduce namespaces and Uniform Resource Identifier (URIs) including their use, advantages and integration into XML
- We will also discuss Regular Expressions as they will become useful in later lessons

Learning outcomes

- Explaining the characteristics of URIs
- Using namespaces and stating their benefits
- Discussing why URIs and namespaces are needed
- Illustrating via examples how URIs are used in XML docs
- Understanding the uses and advantages of Regular Expressions
- Creating basic Regular Expressions

Namespaces, URIs and relation to XML

Namespaces: Motivation

```
<?xml version="1.0" encoding="UTF-8"?>
<Buch>
  <Titel>Semantic Web Grundlagen</Titel>
  <Autor>
    <Name>Pascal Hitzler</Name>
    <Titel>Dr.</Titel>
  </Autor>
  <Autor>
    <Name>York Sure</Name>
    <Titel>Dr.</Titel>
  </Autor>
</Buch>
```

- `Titel` as book title
- `Titel` as academic title
- How can someone correctly interpret `Titel`?
  - From the XML structure and nested levels?
  - From the content on the XML elements?
- It would be better to have a way to "know" rather than "guess".

Namespaces: Examples

- For instance book as namespace for book related stuff
  - Book title: book:Title
  - Book author: book:Author
- People can also have titles
  - Let's use person as namespace for people
    and then we have: person:Title
  - We can also have a namespace for authors: author,
    and then we get author:Title

Namespace: Definition

- A namespace is given a name.
- Namespaces will be related to their domain.
- In a domain, certain words have a common predefined meaning
- Using words/terms from a domain makes it easier:
  - common interpretation and differentiation:
    author:Title versus book:Title
  - reduces ambiguity:
    either author:Title or book:Title (but not both)
  - certainty on the meaning/semantics, people explicitly see author:Title or book:Title

Uniform Resource Identifier

- String of characters acting as an identifier
- Denoting abstract resources, e.g. names or numbers
- Or physical resources, e.g. file, webpage or printer
- Predefined `https://tools.ietf.org/html/rfc3986` syntax rules but still extensible
- A URI is also an Internationalized Resource Identifier IRI

Examples of URIs

- Uniform Resource Locators (URL) — you can directly access them on the Web!
  - Some data on the Web, accessible via `ftp`
    `ftp://ftp.ncbi.nlm.nih.gov/pubmed/baseline`
  - A HTML website
    `http://www.zbmed.de`
  - An email address
    `mailto:info@whitehouse.com`
- An item in a collection (via Uniform Resource Name URN)
  `urn:miriam:pubmed:16333295`

IRI, URI, URL, URN

- IRI $\supset$ URI (all URIs are IRIs)
- URI $\supset$ URL (all URLs are URIs)
- URI $\supset$ URN (all URNs are URIs)
- URN $\cap$ URL $= \varnothing$ (all URLs are URIs)
- Whenever we talk about URIs in this lecture, we mean URIs used on the Web

Structure of URIs (1)

`Schema://[Authorithy][/Path][?Query][#fragment]`

- The schemas `ftp`, `http` or `mailto` correspond to a protocol, used to interpret the URI.
- The `Authority` is the resource provider, commonly a `Host` or `Server`, sometimes with a `Port`
- The `Port` number, allows multiple services from the same server
- The `Path` offers a way toward a particular service or resource provided by the `Authority`
- The `Query` specifies the query string or request sent to the host
- The `#Fragment` directs toward a secondary resource part of the primary one

Structure of URIs (2)

`Schema://[Authorithy][/Path][?Query][#fragment]`

- Conditions to access the server:
  - If the access is not public, the way to get access via registration, membership or payment should be stated
  - Some protocols/schemas allow for authentication in the form of `username:password`

XML namespaces

```
<?xml version="1.0" encoding="UTF-8"?>
<Buch>
  <Titel>Semantic Web Grundlagen</Titel>
  <Autor>
    <Name>Pascal Hitzler</Name>
    <Titel>Dr.</Titel>
  </Autor>
  <Autor>
    <Name>York Sure</Name>
    <Titel>Dr.</Titel>
  </Autor>
</Buch>
```

- `Title` as book title
- `Title` as academic title

XML namespaces: Specification (1)

XML namespaces follow some rules:

- Namespaces can be defined as an element attribute

```
<markup:table xmlns:markup="http://www.w3.org/1999/xhtml/">
  <markup:tr>
    <markup:th>
      Column heading
    </markup:th>
  </markup:tr>
</markup:table>
```

- The namespace name will go right after `xmlns:`

XML namespaces: Specification (2)

But... it is a better idea to define namespaces on the root element

```
<?xml version="1.0" encoding="utf-8"?>
<root xmlns:up="http://purl.uniprot.org/uniprot/"
  xmlns:schema="http://schema.org/" >
  <up:accession>P05067</up:accession>
  <schema:name>APP</schema:name>
```

- as many namespaces as needed
- the namespaces can now be used anywhere in the XML
- easier and shorter than using the full names everywhere

XML namespaces: Specification (3)

If there is a namespace used more than others along the XML, it can be defined as "default" namespace

```
<?xml version="1.0" encoding="utf-8"?>
<uniprot xmlns="http://purl.uniprot.org/uniprot/">
<protein>
  <recommendedName>
    <fullName>Amyloid-beta precursor protein</fullName>
    <shortName>APP</shortName>
</recommendedName>
</uniprot>
```

- note that there is no name after `xmlns` for the base namespace
- no need to mention the base namespace on the XML elements

Regular expressions

Regular Expressions (1)

- Regular expressions are an important means to specify language patterns with a special description language (= regular expression grammar).
- Regular expressions have particularly good-natured characteristics, since it is always fast and efficiently decidable (i.e. in linear processing time) whether a string is correctly formed according to the RegExp grammar.
- The most important datatypes in computer science and also the URIs (but not necessarily the full XML schema) can be specified with RegExp (as it is done for the semantic Web).

Regular Expressions (2)

- Regular Expressions (regexp) are specifications in a special grammar.
- Each specification can be used to find a piece of text and to replace it.
- Example: The word “Text” can be used to search “Text” in the document, but we can also use the specification “[tT]ex[tT]”, to search “Text” as well as “text”, “TexT” and “texT”.
- “[0-9]”: recognizes the numbers from 0 bis 9, but no two-digit numbers yet
- “[1-9][0-9]”: is used for two digit numbers from 10 to 99
- “[0-9]*”: denotes all sequences of numbers, but it may contain one or more zeroes in the beginning of the number.

Composing “regexp”

- All signs in our language are part of regexp: a, b, c, . . . , A, B, Z, . . . , 0, 1, 2, . . . , !, ?, %, $, &, . . .
- “|”: denotes alternatives, e.g. “Text|text|texT|TexT”
- “()”: forms groups, e.g. “(T|t)ex(T|t)”
- “[]”: is used to form groups and ranges, e.g. “[Tt]ext[Tt]” or “[Tt]ext[0-9]”

Composing “regexp” (2)

- “*”: denotes zero or multiple occurrences of a sign, e.g. “Tex[t]*” for Tex, Text, Textt, Textttttttttttt, and others
- “+”: denotes one or multiple occurrences of a sign, e.g. “[Tt]ex[t]+” for Text, Textt, Textttttttttttt
- “?”: denotes zero or one occurrence of a sign, e.g. “[Tt]ex[t]?” for Tex and Text
- Further symbols denote the beginning and the end of a line, i.e. for all signs and others
- From all elements we can form any combination,
  e.g. “[a-zA-Z][:](\[][a-zA-Z][-a-zA-Z0-9]*)+”
  to denote the path of a file in a certain drive.

Summary

Summary

- Namespaces are better defined on the XML header/root so they can be used all along
- Namespaces are useful and simple as they improve readability and make XML more efficient and understandable (no more ambiguity for book or academic title)
- A URI (in the form of a URL) refers to a resource on the web, i.e. electronic data with a unique address
- URIs are useful for identifying resources (and related concepts) and making them usable for computers (and humans)
- URIs are meant to be unique and global but not necessarily persistent (i.e. URIs can disappear)
- Regular Expressions make it easier to find patterns in strings — they can be used to find patterns on URIs

So where are we now?

- For biomedical semantics:
  XML has been used for years to provide data. We have seen some examples on how it is used in UniProt and how URIs and namespaces make XML files easier and simpler to follow (more on XML in Lesson 5)
- For information retrieval and knowledge discovery:
  - URIs are used all along the Web
  - XML is one of the most used formats for retrieval portals to provide data downloads
  - Regular expressions can be combined with term lookup and data retrieval approaches (mentioned in relation to terminologies and language technologies in Lesson 2)