

# ΕΠΛ 602:Foundations of Internet Technologies

The Web of Data (XML, RDF)

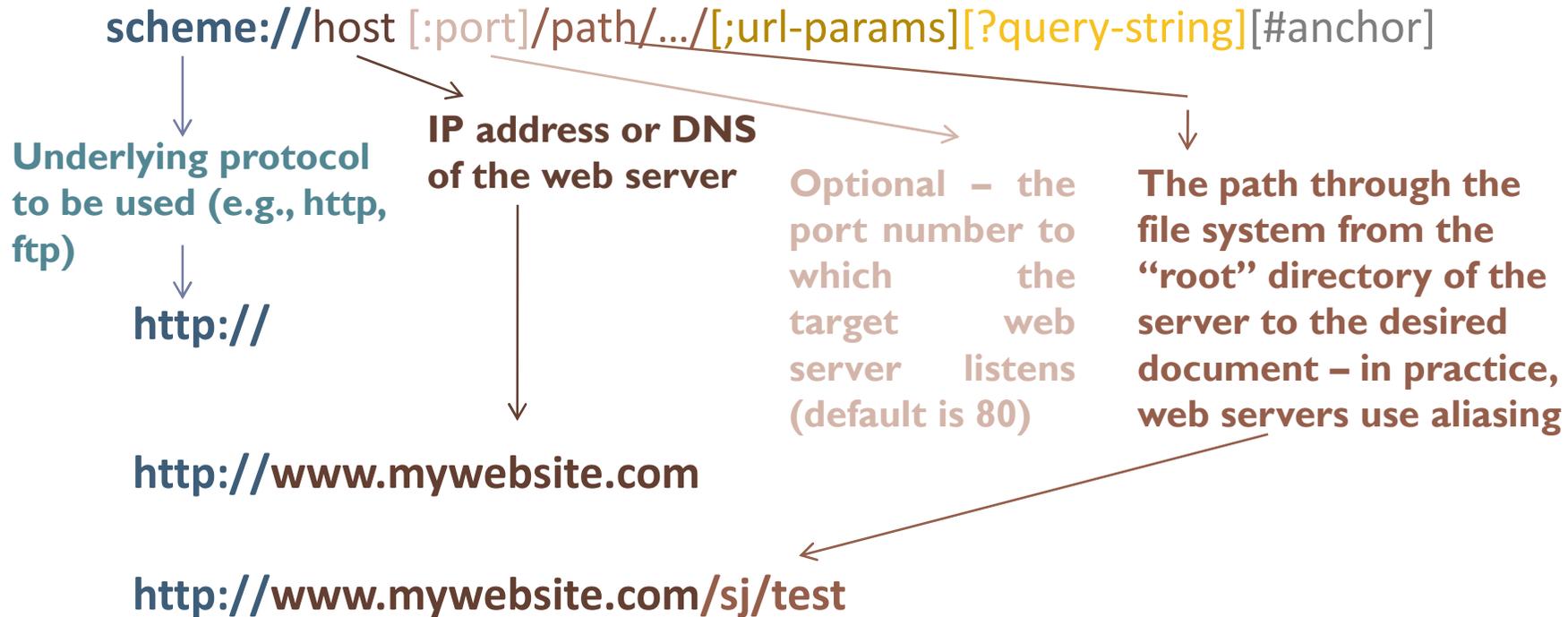
# Lecture Outline

---

- ❖ URL/URI
- ❖ XML (chapter 1, wdm book) , JSON
- ❖ RDF, RDFS, OWL (chapter 7, wdm book)
- ❖ Linked Data

# Uniform Resource Locator (URL)

---



# Uniform Resource Locator (URL)

---

<http://www.mywebsite.com/sj/test>



Optional – name, value pairs; commonly used for session ids in application servers supporting the Java Servlet API

Optional – name, value pairs; for dynamic parameters associated with the request (tracking or context setting, also in HTML forms) passed to the software at the server

Optional – reference to a positional marker within the document

<http://www.mywebsite.com/sj/test?id=8079>

<http://www.mywebsite.com/sj/test?id=8079?name=bob&x=true#label>

`application-protocol://IP-address[:port]path-from-the-root[;par][?dyn-par][#anchor]`

# Status Codes: Redirection (3xx)

---

❖ Redirection: the browser is instructed to resubmit the request to another URL

- 301 moved permanently
- 302 temporarily

at the location specified at the Location header of the response

❖ Browsers respond “silently” to redirection status codes (users unaware)

❖ (not supported or disabled) Web servers include a message body that explicitly references a link to the new location -> follow the link manually (body displayed, if not supported/disabled)

❖ Web servers treat *a URL ending in a slash* as a request for a directory (depending on the server configuration: return either a file with a default name (e.g., index.html) or the contents of the directory)

❖ If the user forgets the trailing “/”, the server a redirection response

❖ Proxies react to 301 status by updating internal relocation tables (*cache 301 redirections*) e.g., redirecting users to the login page when trying to access a protected URL

# Notes

---

**URI** = Uniform Resource Identifier

a string of characters used to identify a name or a resource on the Internet.

By name, by location or both

**Uniform Resource Locator (URL)** is a subset of the Uniform Resource Identifier (URI) that specifies where an identified resource is available and the mechanism for retrieving it.

**URN**

urn:<namespace>:<string>

# Notes

---

**URL shortening:** a technique in which a URL is made substantially shorter achieved by using an HTTP Redirect on a domain name that is short, which links to the web page that has a long URL.

For example,

[http://en.wikipedia.org/wiki/URL\\_shortening](http://en.wikipedia.org/wiki/URL_shortening) ->

<http://bit.ly/urlwiki> or <http://tinyurl.com/urlwiki>.

Convenient for messaging technologies such as [Twitter](#)

In November 2009, the shortened links on one URL shortening service were accessed 2.1 billion times

Normally, a URL shortening service will use the top-level domain of a country that allows foreign sites to use its extension, such as .ly or .to (Libya and Tonga), to redirect worldwide using a short alphanumeric sequence after the provider's site address to point to the long URL.

Another use of URL shortening is to disguise the underlying address.

---

XML

# External Data Representation

---

- ▶ **Person struct in XML**
  - ▶ Tag names: person, name, place, year
  - ▶ Element: <name>Smith</name>
  - ▶ Attribute: id="123456789" of person
  - ▶ Binary data need to be converted to characters (base64)

```
<person id="123456789">  
    <name>Smith</name>  
    <place>London</place>  
    <year>1934</year>  
    <!-- a comment -->  
</person >
```



# External Data Representation: XML namespace

---

- ▶ Name clashes within an application
- ▶ Namespaces: a set of names for a collection of element types and attributes
- ▶ `xmlns:xml` namespace
- ▶ `pers:` name of the name space (used as a prefix)
- ▶ <http://www.cdk4.net/person> : location of schema

```
<person pers:id="123456789" xmlns:pers = "http://www.cdk4.net/person">  
  <pers:name> Smith </pers:name>  
  <pers:place> London </pers:place >  
  <pers:year> 1934 </pers:year>  
</person>
```

---



# External Data Representation: XML schema

---

- ▶ Defines elements and attributes
- ▶ Similar to type definition
- ▶ `xsd:` namespace for xml schema definition

```
<xsd:schema xmlns:xsd = URL of XML schema definitions >
  <xsd:element name= "person" type ="personType" />
  <xsd:complexType name="personType">
    <xsd:sequence>
      <xsd:element name = "name" type="xs:string"/>
      <xsd:element name = "place" type="xs:string"/>
      <xsd:element name = "year" type="xs:positiveInteger"/>
    </xsd:sequence>
    <xsd:attribute name= "id" type = "xs:positiveInteger"/>
  </xsd:complexType>
</xsd:schema>
```

---



# XHTML - stylesheet

---

Separate the content of the document and its layout so as to generate different layouts to present the document (including layouts, colors and fonts)

A stylesheet to provide transformation rules

**Cascading Style Sheets (CSS)** is a style sheet language used for describing the presentation semantics of a document written in a markup language.

- ❖ enable multiple pages to share formatting,
- ❖ reduce complexity and repetition in the structural content (such as by allowing for tableless web design)
- ❖ allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader)
- ❖ allow the web page to display differently depending on the screen size or device

CSS specifies a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called *cascade*, priorities or *weights* are calculated and assigned to rules, so that the results are predictable.

# XHTML - stylesheet

---

A style sheet consists of a list of *rules*.

Each rule or rule-set consists of one or more *selectors* and a *declaration block*.

CSS files are inserted into HTML documents using the following syntax:

```
<link rel="stylesheet" href="http://example.com/css/style.css" type="text/css" />
```

<http://www.w3schools.com/css/default.asp>

```
body
{
background-color:#d0e4fe;
}
h1
{
color:orange;
text-align:center;
}
p
{
font-family:"Times New Roman";
font-size:20px;
}
```

---

RDF

# Standard namespaces

---

rdf:A namespace for RDF.

The URI is: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

rdfs:A namespace for RDFS.

The URI is: <http://www.w3.org/2000/01/rdf-schema#>

owl:A namespace for OWL.

The URI is: <http://www.w3.org/2002/07/owl#>

dc:A namespace for the Dublin Core Initiative.

The URI is: <http://dublincore.org/documents/dcmi-namespace/>

foaf:A namespace for FOAF.

The URI is: <http://xmlns.com/foaf/0.1/>.

## Blank Nodes or Anonymous Resource

---

**Example 3.1** *The following triplets express that Pierre knows someone named “John Smith” wrote a book entitled “Introduction to Java”.*

```
:Pierre foaf:knows _:p
_:p foaf:name "John Smith"
_:p wrote _:b
_:b dc:title "Introduction to Java"
```

The predicates `foaf:knows` and `foaf:name` belong to the FOAF vocabulary. The predicate `dc:title` belongs to the Dublin Core vocabulary.

**Example 3.2** *Consider again the four triplets that we used to express that Pierre knows someone named “John Smith” wrote a book entitled “Introduction to Java”. They are interpreted in FOL by the following positive existential conjunctive formula, where the prefixes (`foaf:`, `dc:`, `_:` and `:`) for denoting the constants, predicates and variables have been omitted for readability.*

$$\exists p \exists b [\text{knows}(\text{Pierre}, p) \wedge \text{name}(p, \text{“John Smith”}) \wedge \text{wrote}(p, b) \wedge \text{title}(b, \text{“Introduction to Java”})]$$

# RDFS: Inference Rules

---

```
if (  $r P s$  ) and (  $P \text{ rdfs:subPropertyOf } Q$  ) then (  $r Q s$  ).  
if (  $P \text{ rdfs:domain } C$  ) and (  $x P y$  ) then (  $x \text{ rdfs:type } C$  ).  
if (  $P \text{ rdfs:range } D$  ) and (  $x P y$  ) then (  $y \text{ rdfs:type } D$  ).
```

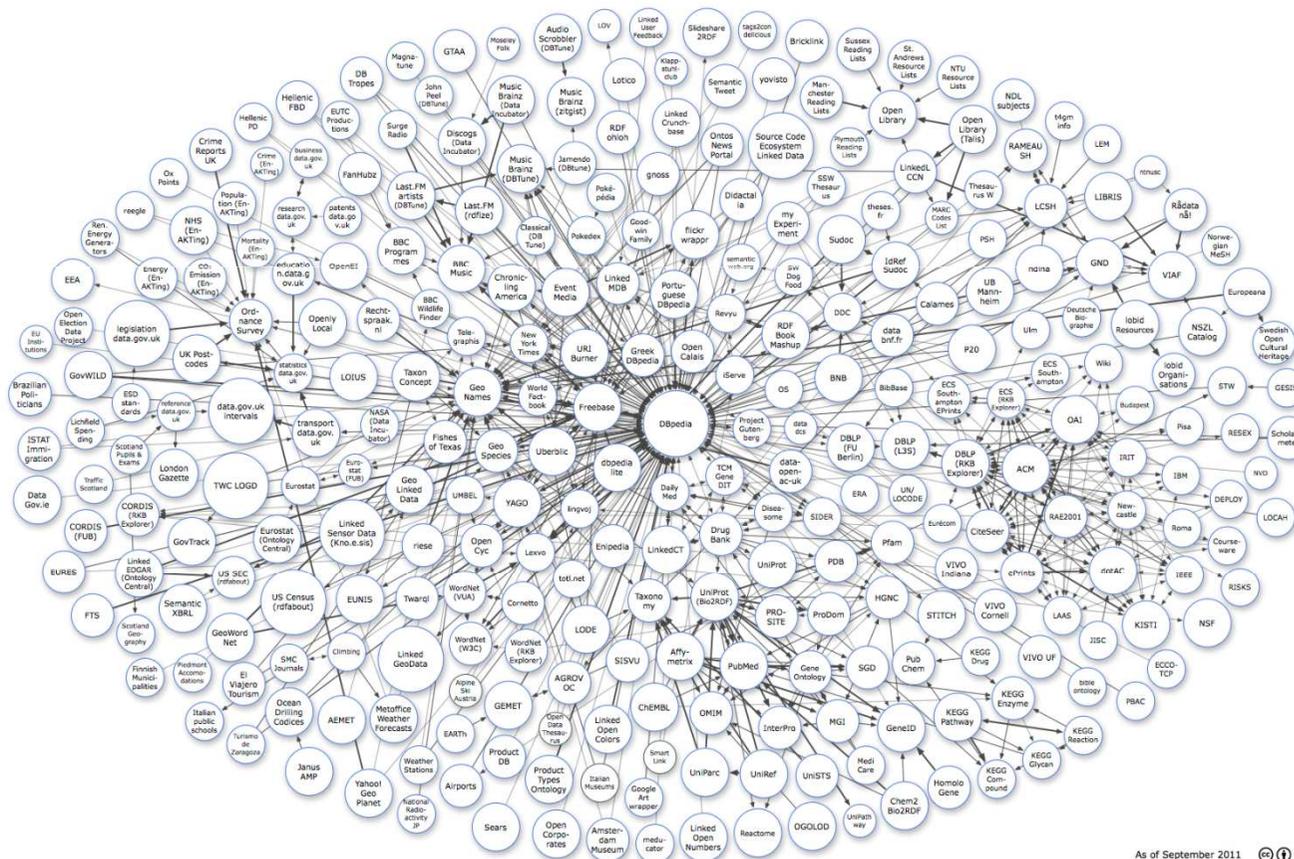
```
if (  $P \text{ rdfs:domain } A$  ) and (  $A \text{ rdfs:subClassOf } B$  )  
  then (  $P \text{ rdfs:domain } B$  )  
if (  $P \text{ rdfs:range } C$  ) and (  $C \text{ rdfs:subClassOf } D$  )  
  then (  $P \text{ rdfs:range } D$  )  
if (  $A \text{ rdfs:subClassOf } B$  ) and (  $B \text{ rdfs:subClassOf } C$  )  
  then (  $A \text{ rdfs:subClassOf } C$  )  
if (  $P \text{ rdfs:subPropertyOf } Q$  ) and (  $Q \text{ rdfs:subPropertyOf } R$  )  
  then (  $P \text{ rdfs:subPropertyOf } R$  )
```

---

# Linked Data

# Linked data

<http://linkeddata.org/>



---

Questions?