XML and JSON Parsing

Why use XML or JSON

We’ve learned how to send arbitrary strings in a variety of ways over HTTP. However, as of yet this has been unstructured text. How do we send representations of objects across different platforms?

XML and JSON to the rescue!

XML (Extensible Markup Language)

XML is formatted in the same manner as HTML except there is no strict dependence on pre-defined tags.

XML vs HTML

While HTML describes how data is formatted, XML defines what the data is itself.

XML should never contain information on how data is styled/formatted.

XML vs HTML

```
<?xml version="1.0"
encoding="UTF-8"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this
weekend!</body>
</note>
```

The above XML describes a note object and each field in it.

```
<!DOCTYPE html>
<html>
<body>
<div>
  <h1>Tove</h1>
  <h1>Jani</h1>
  <h1>Reminder</h1>
  <p>Don't forget me this
weekend!</p>
</div>
```

The above HTML describes how the notes object is formatted but gives us no real useful information on what a notes object is.

XML Rules

- An XML document must have a root node.
  - <note> is the root from the previous example.
- Before the root node is the prolog but this is optional. If it exists it must be the first element.
- In XML multiple white spaces are preserved (In HTML it is not)

More XML Syntax

- XML is Case sensitive.
- XML must have closing tags
  - <myTag/> is not valid in XML.
- Like HTML XML attributes must be in quotes.

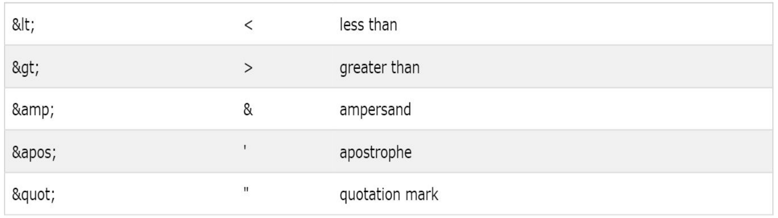
```
<note date="12/11/2007">
  <to>Tove</to>
  <from>Jani</from>
</note>
```

Entity References

Some values like > and < are invalid to be used as values inside of XML since they break parsing. Instead using an entity reference.

```
<message>salary &lt;
1000</message>

<message>salary <
1000</message>
```



&lt;	<	less than
&gt;	>	greater than
&amp;	&	ampersand
&apos;	'	apostrophe
&quot;	"	quotation mark

Comments in XML

```
<!-- This is a comment -->
```

XPath

Xpath is a way of describing the path to an element inside xml using expressions and is supported in most popular languages.

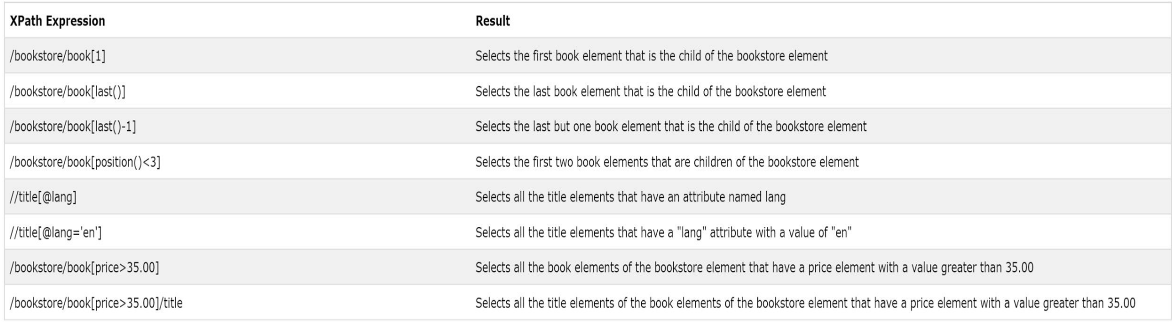
THE UNIVERSITY OF MEMPHIS
Computer Science

XPath Example

```
<bookstore>

<book category="cooking">
  <title lang="en">Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>

<book category="children">
  <title lang="en">Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>
</bookstore>
```



XPath Expression	Result
/bookstore/book[1]	Selects the first book element that is the child of the bookstore element
/bookstore/book[last()]	Selects the last book element that is the child of the bookstore element
/bookstore/book[last()-1]	Selects the last but one book element that is the child of the bookstore element
/bookstore/book[position()<3]	Selects the first two book elements that are children of the bookstore element
//title[@lang]	Selects all the title elements that have an attribute named lang
//title[@lang='en']	Selects all the title elements that have a "lang" attribute with a value of "en"
/bookstore/book[price>35.00]	Selects all the book elements of the bookstore element that have a price element with a value greater than 35.00
/bookstore/book[price>35.00]/title	Selects all the title elements of the book elements of the bookstore element that have a price element with a value greater than 35.00

Marshalling Java Classes to XML

Java provides a class called XMLEncoder that will encode both fields and method headers.

Note that only the method header is described but the implementation of the method is not.

XMLEncoder supports nested objects.

Requirements to use XMLEncoder

- Any instance variable must have a getter and setter in order to be serialized to XML.
- The class must implement the *Serializable* interface.

JSON (Javascript Object Notation)

JSON is an alternative to XML. It's less wordy than xml and can be naturally parsed by any web browser that supports javascript.

JSON vs XML

XML prefers a markup style notation similar to HTML.

JSON prefers name/value pairs.

Both infer the data type by its representation.

ie. “value”:”1” ← string

“value”:1 ← integer

“value”:1.1 ← float

“value”: true ← boolean

Examples

JSON

```
{"employees":[
     {"firstName":"John", "lastName":"Doe"},
     {"firstName":"Anna", "lastName":"Smith"},
     {"firstName":"Peter", "lastName":"Jones"}
]}
```

XML

```
<employees>
    <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
    </employee>
</employees>
```

Parsing JSON in javascript

There are two functions eval() and JSON.parse(). The latter is more secure. Both functions will return javascript objects.

JSON Caveats

- JSON does not support defining methods or functions.
- JSON does not support commenting of any kind.
- Though an XML document is generally larger than the same JSON document, gzip can generally compress them both to the same size.
- JSON does not have the concept of *schema*.
    - Schema are essentially rulesets expressed in XML.

Serializing to JSON

Using a library from json.org we can convert from xml to json easily. Later we will learn how to convert java objects directly without using xml as an itermediary.