XML in a Nutshell

XML, e**X**tended **M**arkup **L**anguage is a collection of rules for universal markup of data.

Brief History

XML has two predecessors:

SGML. SGML stands for Standard Generalized Markup Language. It was designed in late 1980s for document markup. SGML has a large specification and a set of complicated rules for dealing with exceptional cases of document markup. SGML was the first language that used the "angle bracket" syntax for markup elements: ‘‘<a>", ‘‘<b>", etc.

SGML does not define a specific set of markup elements (tags), rather it specifies the rules for creation of markup languages.

HTML. HTML is a specific markup language designed for the purpose of directing the rendering of the content of web pages. Unlike SGML, HTML does have a specific set of tags.

XML combines two important features of these languages:

- Flexibility of SGML: like SGML, XML is a set of rules for defining a specific markup langauge. It allows interested parties to create their own markup language and use it for marking up any data.
- Simplicity of HTML: like HTML, and **unlike** SGML, XML specification is quite simple, and easy-to-understand. XML is much easier to use than SGML (technically, XML is a proper subset of SGML spec.).

The XML specification has been developed and maintained by the World Wide Web Consortium (W3C) (http://www.w3.org). The first *W3C Recommendation* (the official name of the final technical specifications released by W3C) appeared in 1998.

XML Syntax

XML documents consist for the following:

- XML version identification. The first line of any XML document identifies it as an XML document, specifies the XML version it complies to, and identifies any other potential information useful for automated processing of XML documents. A simple first line looks as follows:

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

  The `<?` syntax is used in XML to identify so called *processor instructions*. They are not part of the XML document, rather they are used by XML processors as instructions on how to process the XML[1].

- Comments. Comments in XML have the same syntax as in HTML:

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

- XML markup. XML markup is the parts of the XML document enclosed in angle brackets: `'<a>'`, `'<b id="10">'`, `'<p/>'`, `'</a>'`, etc. A string enclosed in a pair of angle brackets is called a *tag*. There are three types of tags in XML:

  - Opening tags. These tags indicate the beginning of a specific markup. The tag identifier starts immediately after the opening angle bracket.

    ```
    <a>, <myfavoritetag>, <sentence>, <name>
    ```

    are all opening tags. Some extra information, called *attributes* can be stored inside an opening tag:

    ```
    <a href="http://www.cs.uky.edu">, <b id ="10>, <name language="English">
    ```

  - Closing tags. These tags indicate the end of the scope of a specific markup. The closing tag starts with the slash character ("/") immediately after the `''<"`, followed by the tag identifier.

    ```
    </a>, </b>, </sentence>, </name>
    ```

    No attributes are allowed inside the closing tags.

[1] Currently, two versions of XML exist, 1.0 and 1.1. The difference between the two is the version 1.1. provides full support for Unicode documents.

– Empty tags. These tags indicate that the markup does not have scope - it starts and ends at the same place. The closing tag has the tag identifier immediately follow the "¡". Attributes are allowed in the empty tags. The empty tag indicator is the slash character, ''/" right before the closing angle bracket:

```
<a/>  <myemptytag id="10"/>  <name content="John"/>
```

As seen from these examples, XML allows arbitrary tag names in its documents. (For practical purposes you can assume that the tag names are constructed in a manner similar to variable names in Java programs: they start with a letter character, followed by a sequence of zero or more letters or digits or underscores.) *XML is case-sensitive.*

- Content. XML documents are human-readable text documents. The content of XML documents is all the text that is not inside the comments, tags or processing instructions. In the following simple example,

```
<?xml version="1.0"?>
<root>
  This is a test
</root>
```

the string ''This is a test" is the content of the <root> element.

All XML documents must have one XML element, called *the root element* which encompasses all other XML elements and content. Thus, the following document is not XML:

```
 <?xml version="1.0"?>
<a>
  This is a test
</a>
<b>Hello, World!</b>
```

To make it a proper XML document, the entire content must be enclosed in a single element:

```
 <?xml version="1.0"?>
<x>
  <a>
    This is a test
  </a>
  <b>Hello, World!</b>
</x>
```

Well-formed XML documents

All XML documents must be **well-formed.**.

The notion of well-formedness is formally defined in the XML Recommendation. The requirements for well-formed documents are as follows:

1. Correctly placed tags. Each opening tag must have a matching closing tag, and all tags must be properly nested within each other. The following fragment:

   ```
   <x> this is <y> a test</x> !</y>
   ```

   is NOT well-formed.

2. Uniqueness of attribute names. No duplicate attribute names are allowed in the opening and empty tags. The following fragment

   ```
   <a id="10" id="ten">this is a test</a>
   ```

   is NOT well-formed.

3. No ‘‘<" in attribute values. Attribute values should not contain ‘‘<".

The first requirement is the most important and over-arching of all.

Non-well-formed XML documents are not considered.

Valid XML documents

XML documents can come with "schema" descriptions that outline the expected structures to be found in XML documents.

For example, one might want to create XML documents that contain only three types of elements:

- a single <root> element.
- multiple <page> elements, all inside the root element.
- multiple <line> elements, all inside page elements.

There is a number of ways to specify such rules. XML Recommendation includes the description of DTDs, *Document Type Definitions*, which can be embedded into XML documents and used to specify the structure of the XML documents.

In addition to this, W3C has developed XML Schema and XML language for specifying XML schemas, which is more complex than DTD.

Other formalisms, such as a popular *Relax-NG* algebra exist that also can be used.

XML Recommendation describes the requirements for XML document to be valid with respect to a DTD specification. Validity notion is extended to other ways of representing the schema. In a nutshell, validity requirements state that the structure of XML documents must conform to the structure described by the DTD (or other schema description).

XML Parsers

The benefit of using XML lies in the fact, that there is ample available software for parsing and handling it. This allows programmers not to concentrate on parsing XML.

There are two types of XML parsers.

Simple API for XML (SAX)

The first type of parsers, SAX parsers are event-based parsers. They read XML documents as a stream of data and tokenize them into *event messages*. The event message types are:

- opening tag
- closing tag[2]
- text (otherwise known as PCDATA)
- comment
- processing instruction

The stream of events is produced as the output, available for application programs.

Document Object Model Parsers

Document Object Model, DOM, is a tree representation of XML.

DOM XML parsers, accept the stream of SAX events and based on them build a main-memory tree structure called the DOM Tree, the represents the content of the document.

Application programs are provided with the DOM API to access the DOM Tree and navigate its content.

[2] An empty tag `<a/>` is represented as s sequence of two events: `<a></a>`.