**CHAPTER 1: XML**

* XML (eXtensible MarkupLanguage) is a metalanguage(a language used todescribe other languages) for defining vocabularies, which is the key to XML’s importance and popularity.
* XML vocabulary documents are like HTML documents in that they are text-based and consist of mark-up(encoded descriptions of a document’s logical structure) and content(document text not interpreted as markup). Markup is evidenced via tags (angle bracket-delimited syntactic constructs) and each tag has a name.Furthermore, some tags haveattributes.

**Language Features Tour**

XML provides several language features: XML declaration, elements and attributes, character referencesand CDATA sections, namespaces, and comments and processinginstructions

**XML Declaration**

* An XML document usually begins with the XML declaration
* The XML declaration minimally looks like <?xml version="1.0"?> in which the non optional version attribute identifies the version of the XML specification to which the document conforms

**Elements and Attributes**

* A hierarchical (tree) structure of elements,where an element is a portion of the document delimited by a start tag (such as<name>) and an end tag (such as</name>)
* Elements can contain child elements, content, ormixed content(acombination of child elements and content)
* An XML element’s start tag can contain one or more attributes. Attributes provide additional details about elements. Attributes can be optional. When not specified, a defaultvalue of1is assumed.
* Element and attribute names may contain any alphanumeric character from English or another language, and may also include the underscore (\_), hyphen (-), period (.), and colon (:) punctuation characters. The colon should only be used with namespaces (discussed later in this chapter), and names cannot contain whitespace.

**Character References and CDATA Section**

* Certain characters cannot appear literally in the content that appears between a start tag and an end tag or within an attribute value.
* One solution to this problem is to replace the literal character with acharacter reference, which is a code that represents the character
* Characterreferences are classified as:
* Anumeric character referencerefers to a charactervia its Unicode code point and adheres to the format&#nnnn;(not restricted to four positions) or&#xhhhh;(not restricted to four positions), wherennnnprovidesa decimal representation of the code point andhhhhprovides a hexadecimal representation.
* Acharacter entity referencerefers to a character viathe name of anentity(aliased data) that specifies thedesired character as its replacement text
* ACDATA sectionis a section of literal HTML or XML markup and content surrounded by the <![CDATA[prefix and the]]> suffix. You don’t need to specify predefined character entity references within a CDATA section,

**Namespace**

* It’s common to create XML documents that combine features from differentXML languages. Namespaces are used to prevent name conflicts whenelements and other XML language features appea
* Without namespaces, anXML parser couldn’t distinguish between same-named elements or other language features that mean different thing
* A namespace is a Uniform Resource Identifier (URI)-based container that helps differentiate XML vocabularies by providing a unique context for its contained identifiers.
* The namespace URI is associated with a name space prefix (an alias for the URI). When prefix is specified, the prefix and a colon character are prepended to the name of each element tag that belongs to that namespace
* All element tags that associate with XHTML are prefixed with h: and all element tags that associate with the recipe language are prefixed with r:

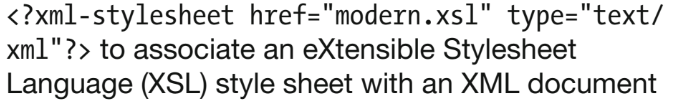


* A tag’s attributes don’t need to be prefixed when those attributes belong to the element. However, a prefix is required for attributes belonging to other namespaces.
* The XHTML style attribute has been prefixed with h: because this attribute belongs to the XHTML language namespace and not to the recipe language namespace.



Comments and Processing Instructions

* XML documents can containcomments, which are character sequences beginning with <!-- and ending with -->
* Comments are used to clarify portions of a document. They can appear any where after the XML declaration except within tags, cannot be nested,cannot contain a double hyphen (--)
* XML also permits processing instructions to be present. A processing instructionis an instruction that’s made available to the application parsing the document
* The instruction begins with <? and ends with?>. The <? Prefix is followed by a name known as the target. This name typically identifies the application to which the processing instruction is intended. The rest of the processing instruction contains text in a format appropriate to the application
* Example:



**Well-Formed Documents**

To make XML documents easier to parse, XML mandates that XML documents follow certain rules:

* All elements must either have start and end tags orconsist of empty-element tags
* Tags must be nested correctly.
* All attribute values must be quoted.
* Empty elements must be properly formatted
* Be careful with case

XML parsers that are aware of namespaces enforce two additional rules:

* Each element and attribute name must not include morethan one colon character.
* No entity names, processing instruction targets, ornotation names (discussed later) can contain colons.

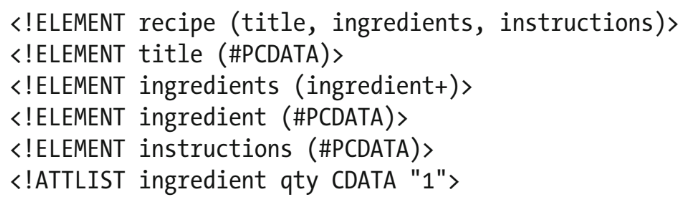
An XML document that conforms to these rules iswell formed. Thedocument has a logical and clean appearance and is much easier toprocess. XML parsers will only parse well-formed XML documents

**Valid Documents**

* A valid document adheres to constraints.
* Some XML parsers perform validation, whereas other parsers don’t because validating parsers are harder to write. A parser that performs validation compares an XML document to a grammar document. Any deviation from the grammar document is reported as an error to the application
* Grammar documents are written in a special language. Two commonly used grammar languages are Document Type Definition and XML Schema

**Document Type Definition**

* Document Type Definition (DTD) is the oldest grammar language fors pecifying an XML document’s grammar



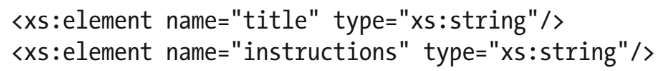
* This DTD first declares the recipe language’s elements. Element declarations take the form <!ELEMENT name content-specifier>, where nameis any legal XML name and content-specifier identifies what can appear within the element
* The first element declaration states that exactly one recipe element can appear in the XML document—this declaration doesn’t imply that recipe is the root element
* The second element declaration states that the title element contains parsed character data (nonmarkup text). The third element declaration states that at least one ingredient element must appear in ingredients
* A DTD-based validating XML parser requires that a document include a document type declaration identifying the DTD that specifies the document’s grammar before it will validate the document
* You can also declare notations and general and parameter entities within DTDs. A notation is an arbitrary piece of data that typically describes the format of unparsed binary data. It’s also common to use notations to specify binary data types via media types
* General entities are entities referenced from inside an XML document via general entity references, syntactic constructs of the form&name. General entities are classified as internal or external. An internal general entity is a general entity whose value is stored in the DTD. An external general entity is a general entity whose value is stored outside the DTD. The value might be textual data (such as an XML document) or it might be binary data (such as a JPEG image).
* An external parsed general entity references an external file that stores the entity’s textual data, which is subject to being inserted into a document and parsed by a validating parser when a general entity reference is specified in the document, and which has the form <!ENTITYnameSYSTEMuri>
* An external unparsed general entityreferences an external file that stores thee ntity’s binary data and has the form <!ENTITYnameSYSTEMuriNDATAnname>, where name identifies the entity,uri locates the external file, and NDATA identifies the notation declaration named nname
* Parameter entities are entities referenced from inside a DTD via parameter entity references, syntactic constructs of the form %name.
* Parameter entities are classified as internal or external. An internal parameter entity is a parameter entity whose value is stored in the DTD and has the form <!ENTITY %name value>, where name identifies the entity and value specifies its value. Anexternal parameter entityis a parameter entity whose value is storedoutside the DTD. It has the form<!ENTITY %name SYSTEM uri>, where name identifies the entity and uri locates the external file

**XML Schema**

* XML Schema is a grammar language for declaring the structure, content, and semantics (meaning) of an XML document. This language’s grammar documents are known as schemas that are themselves XML documents
* XML Schema provides restriction (reducing the set of permitted values through constraints), list (allowing a sequence of values), and union (allowing a choice of values from several types) derivation methods for creating new simple types from these primitive types.
* Introductory schema element:



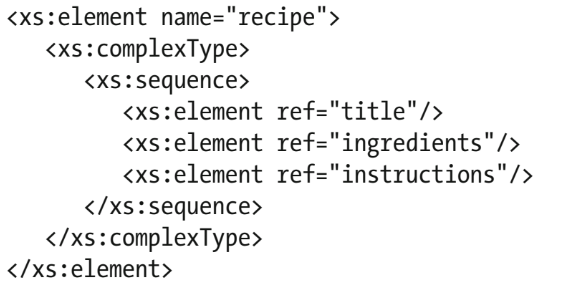
* Use the element element to declare the title and instructions simple type elements, as follows



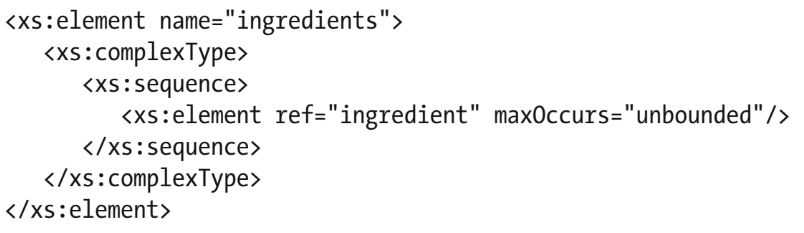
* XML Schema requires that each element have a name and (unlike DTD) be associated with a type, which identifies the kind of data stored in the element
* Use the attribute element to declare the qty simple type attribute, as follows



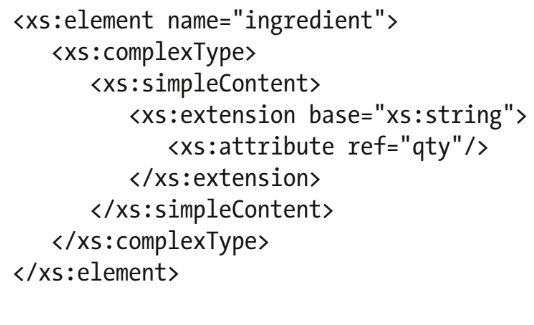
* Declare recipe as follows



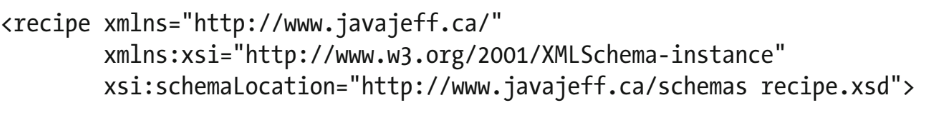
* Declare ingredients:



* Declare ingredient:



* After creating the schema, you can reference it from a recipe document. Accomplish this task by specifying xmlns:xsi and xsi:schemaLocationattributes on the document’s root element start tag (<recipe>), as follows:



* If an XML document declares a namespace (xmlnsdefault orxmlns:prefix),that namespace must be made available to the schema so that a validatingparser can resolve all references to elements and other schema componentsfor that namespace