XML Interview Questions with Answers:

31. What’s a Document Type Definition (DTD) and where do

I get one?

A DTD is a description in XML Declaration Syntax of a particular

type or class of document. It sets out what names are to be used

for the different types of element, where they may occur, and how

they all fit together. (A question C.16, Schema does the same thing

in XML Document Syntax, and allows more extensive data-checking.)

For example, if you want a document type to be able to describe

Lists which contain Items, the relevant part of your DTD might contain

something like this:

&lt;!ELEMENT List (Item)+&gt;

&lt;!ELEMENT Item (#PCDATA)&gt;

This defines a list as an element type containing one or more items

(that’s the plus sign); and it defines items as element types containing

just plain text (Parsed Character Data or PCDATA). Validators read

the DTD before they read your document so that they can identify

where every element type ought to come and how each relates to the

other, so that applications which need to know this in advance (most

editors, search engines, navigators, and databases) can set themselves

up correctly. The example above lets you create lists like:

&lt;List&gt;

&lt;Item&gt;Chocolate&lt;/Item&gt;

&lt;Item&gt;Music&lt;/Item&gt;

&lt;Item&gt;Surfingv&lt;/Item&gt;

&lt;/List&gt;

(The indentation in the example is just for legibility while editing:

it is not required by XML.)

A DTD provides applications with advance notice of what names and

structures can be used in a particular document type. Using a DTD

and a validating editor means you can be certain that all documents

of that particular type will be constructed and named in a consistent

and conformant manner.

DTDs are not required for processing the tip in question Bwell-formed

documents, but they are needed if you want to take advantage of

XML’s special attribute types like the built-in ID/IDREF cross-reference

mechanism; or the use of default attribute values; or references

to external non-XML files (’Notations’); or if you simply

want a check on document validity before processing.

There are thousands of DTDs already in existence in all kinds of

areas (see the SGML/XML Web pages for pointers). Many of them can

be downloaded and used freely; or you can write your own (see the

question on creating your own DTD. Old SGML DTDs need to be converted

to XML for use with XML systems: read the question on converting

SGML DTDs to XML, but most popular SGML DTDs are already available

in XML form.

The alternatives to a DTD are various forms of question C.16, Schema.

These provide more extensive validation features than DTDs, including

character data content validation.

32. Does XML let me make up my own tags?

No, it lets you make up names for your own element types. If you

think tags and elements are the same thing you are already in considerable

trouble: read the rest of this question carefully.

33. How do I create my own document type?

Document types usually need a formal description, either a DTD

or a Schema. Whilst it is possible to process well-formed XML documents

without any such description, trying to create them without one

is asking for trouble. A DTD or Schema is used with an XML editor

or API interface to guide and control the construction of the document,

making sure the right elements go in the right places.

Creating your own document type therefore begins with an analysis

of the class of documents you want to describe: reports, invoices,

letters, configuration files, credit-card verification requests,

or whatever. Once you have the structure correct, you write code

to express this formally, using DTD or Schema syntax.

34. How do I write my own DTD?

You need to use the XML Declaration Syntax (very simple: declaration

keywords begin with

&lt;!ELEMENT Shopping-List (Item)+&gt;

&lt;!ELEMENT Item (#PCDATA)&gt;

It says that there shall be an element called Shopping-List and

that it shall contain elements called Item: there must be at least

one Item (that’s the plus sign) but there may be more than one.

It also says that the Item element may contain only parsed character

data (PCDATA, ie text: no further markup).

Because there is no other element which contains Shopping-List,

that element is assumed to be the ‘root’ element, which

encloses everything else in the document. You can now use it to

create an XML file: give your editor the declarations:

&lt;?xml version=”1.0″?&gt;

&lt;!DOCTYPE Shopping-List SYSTEM “shoplist.dtd”&gt;

(assuming you put the DTD in that file). Now your editor will let

you create files according to the pattern:

&lt;Shopping-List&gt;

&lt;Item&gt;Chocolate&lt;/Item&gt;

&lt;Item&gt;Sugar&lt;/Item&gt;

&lt;Item&gt;Butter&lt;/Item&gt;

&lt;/Shopping-List&gt;

It is possible to develop complex and powerful DTDs of great subtlety,

but for any significant use you should learn more about document

systems analysis and document type design. See for example Developing

SGML DTDs: From Text to Model to Markup (Maler and el Andaloussi,

1995): this was written for SGML but perhaps 95% of it applies to

XML as well, as XML is much simpler than full SGML—see the

list of restrictions which shows what has been cut out.

Warning

Incidentally, a DTD file never has a DOCTYPE Declaration in it:

that only occurs in an XML document instance (it’s what references

the DTD). And a DTD file also never has an XML Declaration at the

top either. Unfortunately there is still software around which inserts

one or both of these.

35. Can a root element type be explicitly declared in the

DTD?

No. This is done in the document’s Document Type Declaration, not

in the DTD.

36. I keep hearing about alternatives to DTDs. What’s a

Schema?

The W3C XML Schema recommendation provides a means of specifying

formal data typing and validation of element content in terms of

data types, so that document type designers can provide criteria

for checking the data content of elements as well as the markup

itself. Schemas are written in XML Document Syntax, like XML documents

are, avoiding the need for processing software to be able to read

XML Declaration Syntax (used for DTDs).

There is a separate Schema FAQ at http://www.schemavalid.com.

The term ‘vocabulary’ is sometimes used to refer to

DTDs and Schemas together. Schemas are aimed at e-commerce, data

control, and database-style applications where character data content

requires validation and where stricter data control is needed than

is possible with DTDs; or where strong data typing is required.

They are usually unnecessary for traditional text document publishing

applications.

Unlike DTDs, Schemas cannot be specified in an XML Document Type

Declaration. They can be specified in a Namespace, where Schema-aware

software should pick it up, but this is optional:

&lt;invoice id=”abc123″

xmlns=”http://example.org/ns/books/”

xmlns:xsi=”http://www.w3.org/2001/XMLSchema-instance”

xsi:schemaLocation=”http://acme.wilycoyote.org/xsd/invoice.xsd”&gt;

…

&lt;/invoice&gt;

More commonly, you specify the Schema in your processing software,

which should record separately which Schema is used by which XML

document instance.

In contrast to the complexity of the W3C Schema model, Relax NG

is a lightweight, easy-to-use XML schema language devised by James

Clark (see http://relaxng.org/) with development hosted by OASIS.

It allows similar richness of expression and the use of XML as its

syntax, but it provides an additional, simplified, syntax which

is easier to use for those accustomed to DTDs.

37. How do I get XML into or out of a database?

Ask your database manufacturer: they all provide XML import and

export modules to connect XML applications with databases. In some

trivial cases there will be a 1:1 match between field names in the

database table and element type names in the XML Schema or DTD,

but in most cases some programming will be required to establish

the desired match. This can usually be stored as a procedure so

that subsequent uses are simply commands or calls with the relevant

parameters.

In less trivial, but still simple, cases, you could export by writing

a report routine that formats the output as an XML document, and

you could import by writing an XSLT transformation that formatted

the XML data as a load file.

38. Can I encode mathematics using XML?Updated

Yes, if the document type you use provides for math, and your users’

browsers are capable of rendering it. The mathematics-using community

has developed the MathML Recommendation at the W3C, which is a native

XML application suitable for embedding in other DTDs and Schemas.

It is also possible to make XML fragments from other DTDs, such

as ISO 12083 Math, or OpenMath, or one of your own making. Browsers

which display math embedded in SGML existed for many years (eg DynaText,

Panorama, Multidoc Pro), and mainstream browsers are now rendering

MathML. David Carlisle has produced a set of stylesheets for rendering

MathML in browsers. It is also possible to use XSLT to convert XML

math markup to LATEX for print (PDF) rendering, or to use XSL:FO.

Please note that XML is not itself a programming language, so concepts

such as arithmetic and if-statements (if-then-else logic) are not

meaningful in XML documents.

39. How will XML affect my document links?

The linking abilities of XML systems are potentially much more

powerful than those of HTML, so you’ll be able to do much more with

them. Existing href-style links will remain usable, but the new

linking technology is based on the lessons learned in the development

of other standards involving hypertext, such as TEI and HyTime,

which let you manage bidirectional and multi-way links, as well

as links to a whole element or span of text (within your own or

other documents) rather than to a single point. These features have

been available to SGML users for many years, so there is considerable

experience and expertise available in using them. Currently only

Mozilla Firefox implements XLink.

The XML Linking Specification (XLink) and the XML Extended Pointer

Specification (XPointer) documents contain the details. An XLink

can be either a URI or a TEI-style Extended Pointer (XPointer),

or both. A URI on its own is assumed to be a resource; if an XPointer

follows it, it is assumed to be a sub-resource of that URI; an XPointer

on its own is assumed to apply to the current document (all exactly

as with HTML).

An XLink may use one of #, ?, or |. The # and ? mean the same as

in HTML applications; the | means the sub-resource can be found

by applying the link to the resource, but the method of doing this

is left to the application. An XPointer can only follow a #.

The TEI Extended Pointer Notation (EPN) is much more powerful than

the fragment address on the end of some URIs, as it allows you to

specify the location of a link end using the structure of the document

as well as (or in addition to) known, fixed points like IDs. For

example, the linked second occurrence of the word ‘XPointer’

two paragraphs back could be referred to with the URI (shown here

with linebreaks and spaces for clarity: in practice it would of

course be all one long string):

http://xml.silmaril.ie/faq.xml#ID(hypertext)

.child(1,#element,’answer’)

.child(2,#element,’para’)

.child(1,#element,’link’)

This means the first link element within the second paragraph within

the answer in the element whose ID is hypertext (this question).

Count the objects from the start of this question (which has the

ID hypertext) in the XML source:

1. the first child object is the element containing the question

();

2. the second child object is the answer (the element);

3. within this element go to the second paragraph;

4. find the first link element.

Eve Maler explained the relationship of XLink and XPointer as follows:

XLink governs how you insert links into your XML document, where

the link might point to anything (eg a GIF file); XPointer governs

the fragment identifier that can go on a URL when you’re linking

to an XML document, from anywhere (eg from an HTML file).

[Or indeed from an XML file, a URI in a mail message, etc…Ed.]

David Megginson has produced an xpointer function for Emacs/psgml

which will deduce an XPointer for any location in an XML document.

XML Spy has a similar function.

40. How does XML handle metadata?

Because XML lets you define your own markup languages, you can

make full use of the extended hypertext features of XML (see the

question on Links) to store or link to metadata in any format (eg

using ISO 11179, as a Topic Maps Published Subject, with Dublin

Core, Warwick Framework, or with Resource Description Framework

(RDF), or even Platform for Internet Content Selection (PICS)).

There are no predefined elements in XML, because it is an architecture,

not an application, so it is not part of XML’s job to specify how

or if authors should or should not implement metadata. You are therefore

free to use any suitable method. Browser makers may also have their

own architectural recommendations or methods to propose.