**In HTML**  
On Web servers, the path to a document is usually found within the URI (or URL) for that document. For example, this page is located at:  
http://webdesign.about.com/library/weekly/aa110501.htm  
The path (not including the file name itself) is indicated in blue, "/library/weekly/".

If I wanted to link to this document from another document, I could write the entire URI, but most likely, I would just use the base path, starting with a slash "/". The preliminary slash tells the browser to start at the document root and move down it through the specified directories to find the document defined:  
/library/weekly/aa110501.htm

**In XPath**  
XPath works in much the same way, only instead of navigating a Web server, it is navigating an XML document. In all XML documents, there must be a root element, this is represented by the slash "/" in XPath. In a standard XHTML document, the root element would be "html", so to match everything in the XHTML document you would write:  
/html  
To match all the paragraphs in an XHTML document, you would write:  
/html/body/p

/html/body/p would match all paragraphs within the body tag, but if there were paragraphs within a div tag or a td tag, these would be skipped. With XPath, you can specify all paragraph tags with two preceding slashes:  
//p  
This would match every paragraph tag in the XHTML document, no matter where it was.  
Or you could match only the paragraph tags that are inside a div tag:  
//div/p

**Selecting Multiple Elements**  
Using the star (\*) selects every element that is within the preceding path. So if you wanted to match every element that is within a td tag (such as p, div, etc.), you would write:  
//td/\*

But you can also use the star to match the preceding path. For example, if you wanted to match every paragraph that was four levels in (such as: /html/body/div/p): /\*/\*/\*/p

**What About Attributes**  
Attributes on a tag can be matched using XPath. You use the at-symbol (@) to match an attribute.

* You can match either the attribute:  
  //@class  
  That matches the attribute "class" in any tag in the document.
* Or you can match the tag with a specific attribute.   
  //p[@class]  
  Matching any paragraph tag with a class attribute.
* Using the star, you can match any paragraph with an attribute, no matter what it is:  
  //p[@\*]  
  This matches any paragraph with an attribute of any type, but skips other paragraph tags.
* You can even match the contents of the attribute:  
  //p[@class='red']  
  This would only match those paragraphs with the attribute and value class="red". Paragraphs with class="blue" would be skipped.

**Advanced XPath**  
There are many additional features of XPath:

* count()  
  Select elements based on a count of their children, such as to get all elements with 3 children
* name()  
  Select elements based on their name, such as to get all elements with "e" in the name
* string-length()  
  Select elements that have a specfic length string, such as to get all elements with names that are longer than 4 letters
* boolean strings  
  Select elements based on a boolean operation, such as all elements that are not named "p" and have an attribute "class"

If you are going to work with XSLT or XPointer, then you should be familiar with the rules of XPath. XPath is a powerful tool to indicate very precise locations within your XML documents.