Дараах XML технологиудыг судлан жишээ дээр ажиллаж тайланг бичнэ.

1. XML

2. DTD

3. XSD

4. XSLT

5. XQuery

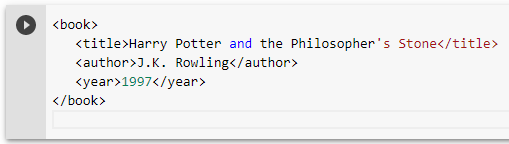
6. XPath

<https://www.tutorialspoint.com/xml/xml_dtds.htm>

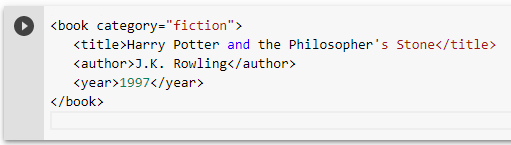
1. XML (Extensible Markup Language)

XML нь өгөгдлийг тодорхойлоход ашигладаг тэмдэглэгээний хэл юм. Энэ нь баримт бичигт өгөгдлийн элементүүдийн бүтэц, утгыг тодорхойлоход хаягуудыг ашигладаг. XML нь янз бүрийн системүүдийн хооронд өгөгдөл солилцохын тулд вэб програмуудад ихэвчлэн ашиглагддаг.

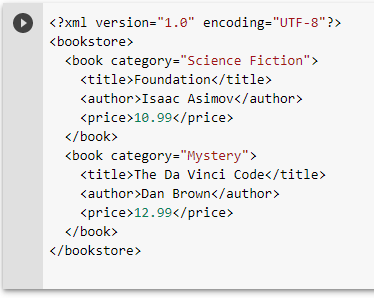
1. Markup language /Тэмдэглэгээний хэл/: XML нь тэмдэглэгээний хэл бөгөөд энэ нь өгөгдлийн бүтэц, утгыг тодорхойлохдоо шошго болон бусад тэмдэглэгээг ашигладаг гэсэн үг юм. Шошгууд нь өгөгдлийн өөр өөр хэсгүүдийг тэмдэглэж, тэдгээрийг хэрхэн тайлбарлах талаар мэдээлэл өгөхөд ашиглагддаг
2. Elements/Элементүүд/: XML нь XML баримт бичгийн үндсэн блок болох элементүүдийг тодорхойлохдоо хаягуудыг ашигладаг. Элемент нь <element>content</element> гэх мэт эхлэл, контент болон төгсгөлийн тагаас бүрдэнэ. Эхлэх шошго нь элементийн эхлэлийг, төгсгөлийн шошго нь элементийн төгсгөлийг, агуулга нь элемент дотор агуулагдах өгөгдөл юм.
3. Tags: XML дээрх шошго нь өнцгийн хаалтанд (< >) хавсаргасан бөгөөд XML баримт бичгийн элементүүдийг тодорхойлдог. Жишээлбэл, дараах XML кодын <book> таг нь элементийг тодорхойлдог:



1. Attributes / Шинж чанарууд /: Элементүүдээс гадна XML нь элементүүдийн талаар нэмэлт мэдээлэл өгөхийн тулд шинж чанаруудыг ашигладаг. Шинж чанаруудыг элементийн эхлэл шошгонд зааж өгсөн бөгөөд элементийн тухай мета өгөгдлийг өгдөг. Жишээлбэл, дараах XML кодын категорийн шинж чанар нь номын элементийн талаар нэмэлт мэдээллийг өгдөг:



1. Content /Агуулга/: XML элементийн агуулга нь эхлэл ба төгсгөлийн шошгонд агуулагдах өгөгдөл юм. Энэ нь текст болон бусад байж болно



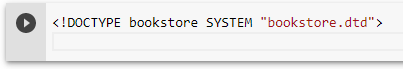
1. The XML declaration / XML мэдэгдэл /: Кодын эхний мөр нь ашиглаж буй XML хувилбар (энэ тохиолдолд 1.0 хувилбар) болон ашиглагдаж буй тэмдэгтийн кодчилол (энэ тохиолдолд UTF-8) -ийг заадаг XML мэдэгдэл юм.
2. The root element / Үндсэн элемент /: Үндсэн элемент нь XML баримт бичгийн дээд түвшний элемент бөгөөд бусад бүх элементүүдийг агуулна. Энэ жишээнд үндсэн элемент нь "номын дэлгүүр" юм.
3. Child elements: "Номын дэлгүүр" элемент нь "ном" гэсэн хоёр хүүхэд элементтэй. "Ном" элемент бүр нь тодорхой номын тухай мэдээллийг агуулдаг. The "bookstore" element has two child elements, "book". Each "book" element contains information about a specific book.
4. Attributes /шинж чанар/: The "book" elements have a "category" attribute that provides additional information about the book's category. "Ном" элементүүд нь номын ангиллын талаар нэмэлт мэдээлэл өгөх "ангилал" шинж чанартай байдаг.
5. Text content: Within each "book" element, there are four child elements: "title", "author", "year", and "price". These elements contain text content that provides information about the book. "Ном" элемент бүрийн дотор "гарчиг", "зохиогч", "жил", "үнэ" гэсэн дөрвөн child элемент байдаг. Эдгээр элементүүд нь номын талаархи мэдээллийг агуулсан text content болдог.
6. Attribute values /утгууд/: The "category" attribute has two possible values, "web" and "fiction", which provide information about the book's category. "Ангилал" шинж чанар нь "вэб" ба "зөгнөлт" гэсэн хоёр боломжит утгатай бөгөөд энэ нь номын ангиллын талаар мэдээлэл өгдөг.

Дүгнэж хэлэхэд XML нь шошго, шинж чанар, контентыг ашиглан өгөгдлийг тодорхойлоход ашигладаг тэмдэглэгээний хэл юм. Энэ нь хүний ​​унших форматаар нарийн төвөгтэй өгөгдлийн бүтцийг илэрхийлж чаддаг тул өгөгдөл солилцох, хадгалахад өргөн хэрэглэгддэг стандарт юм.

1. DTD (Document Type Definition):

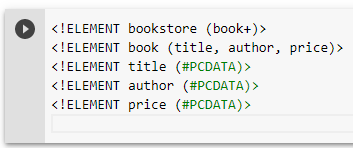
DTD нь XML баримт бичгийн бүтцийг тодорхойлоход хэрэглэгддэг схемийн хэлний нэг төрөл юм. Энэ нь XML баримт бичигт ашиглагдах элементүүд, шинж чанарууд, нэгжүүд, тэдгээрийн харилцаа холбоо, дарааллыг тодорхойлдог. DTD нь XML баримт бичиг нь тодорхой бүтэц эсвэл форматтай нийцэж байгаа эсэхийг баталгаажуулахад ашиглагддаг.

DTD нь ихэвчлэн .dtd өргөтгөлтэй тусдаа файлд тодорхойлогддог бөгөөд DOCTYPE мэдэгдлийг ашиглан XML баримт бичигт лавлаж болно. DTD-д хамаарах DOCTYPE мэдэгдлийн жишээ:

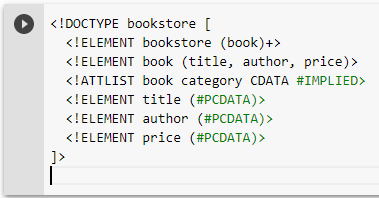


Энэ жишээнд DOCTYPE мэдэгдэл нь XML баримт бичиг нь "bookstore.dtd" файлд тодорхойлогдсон "номын дэлгүүр" DTD-тэй тохирч байгааг зааж өгсөн.

Номын дэлгүүрийн XML баримт бичгийн бүтцийг тодорхойлсон DTD-ийн жишээ:



Энэ жишээнд DTD нь XML баримт бичгийн үндсэн элемент нь "номын дэлгүүр" байх ёстой бөгөөд нэг буюу хэд хэдэн "ном" элемент агуулсан байх ёстойг зааж өгсөн. "Ном" элемент бүр "гарчиг", "зохиогч", "үнэ" гэсэн дарааллаар байх ёстой. "#PCDATA" түлхүүр үг нь тухайн элемент нь задалсан тэмдэгтийн өгөгдлийг агуулж болохыг харуулж байна.

DTD нь харьцангуй энгийн схемийн хэл бөгөөд үүнийг ихэвчлэн жижиг, энгийн XML баримт бичигт ашигладаг. Гэсэн хэдий ч илүү том, илүү төвөгтэй баримт бичгийн хувьд XML схемийг (XSD) илүү уян хатан, ажиллагаатай тул ихэвчлэн илүүд үздэг.

1. **<!DOCTYPE bookstore [** - This is the DOCTYPE declaration that defines the document type as "bookstore". Энэ нь баримт бичгийн төрлийг "номын дэлгүүр" гэж тодорхойлсон DOCTYPE мэдэгдэл юм
2. **<!ELEMENT bookstore (book)+>** - This line defines the "bookstore" element as having one or more "book" elements. The plus sign (+) indicates that the "book" element must appear at least once. Энэ мөрөнд "номын дэлгүүр"-ийн элементийг нэг буюу хэд хэдэн "ном" элементтэй гэж тодорхойлдог. Нэмэх тэмдэг (+) нь "ном" элемент дор хаяж нэг удаа гарч ирэх ёстойг харуулж байна.
3. **<!ELEMENT book (title, author, price)>** - This line defines the "book" element as containing a "title", "author", and "price" element in that order. Энэ мөрөнд "ном" элементийг "гарчиг", "зохиогч", "үнэ" гэсэн дарааллаар нь агуулсан элементийг тодорхойлно.
4. **<!ATTLIST book category CDATA #IMPLIED>** - This line defines an attribute called "category" for the "book" element. The attribute type is CDATA (character data) and the #IMPLIED keyword indicates that the attribute is optional. Энэ мөрөнд "ном" элементийн "категори" хэмээх атрибутыг тодорхойлно. Атрибутын төрөл нь CDATA (тэмдэгтийн өгөгдөл) бөгөөд #IMPLIED түлхүүр үг нь атрибут нь сонголттой гэдгийг илтгэнэ.
5. **<!ELEMENT title (#PCDATA)>** - This line defines the "title" element as containing parsed character data only. The #PCDATA keyword stands for parsed character data. Энэ мөрөнд "гарчиг" элементийг зөвхөн задлан шинжилсэн тэмдэгтийн өгөгдөл агуулсан гэж тодорхойлдог. #PCDATA түлхүүр үг нь задалсан тэмдэгтийн өгөгдлийг илэрхийлдэг.
6. **<!ELEMENT author (#PCDATA)>** - This line defines the "author" element as containing parsed character data only. Энэ мөрөнд "зохиогчийн" элементийг зөвхөн задлан шинжилсэн тэмдэгтийн өгөгдөл агуулсан гэж тодорхойлдог.
7. **<!ELEMENT price (#PCDATA)>** - This line defines the "price" element as containing parsed character data only. Энэ мөрөнд "үнэ" элементийг зөвхөн задлан шинжилсэн тэмдэгтийн өгөгдлийг агуулна гэж тодорхойлдог.

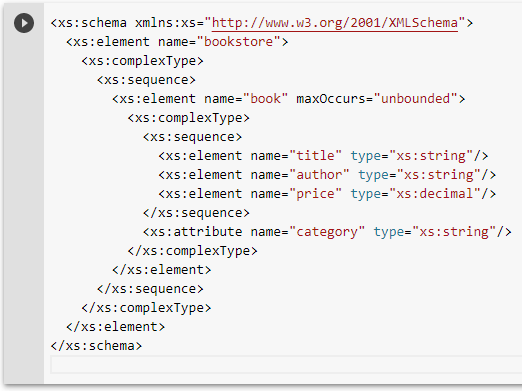
Ерөнхийдөө энэ DTD код нь номын дэлгүүрт зориулсан XML баримт бичгийн бүтцийг тодорхойлдог. "Номын дэлгүүр" элемент нь нэг буюу хэд хэдэн "ном" элементийг агуулсан байх ёстой бөгөөд "ном" элемент бүр "гарчиг", "зохиогч", "үнэ" гэсэн дарааллаар байх ёстой. Нэмж дурдахад, "ном" элемент нь нэмэлт "категори" шинж чанартай байж болно. "Гарчиг", "зохиогч", "үнэ" элементүүд нь зөвхөн задлан шинжлэгдсэн тэмдэгтийн өгөгдлийг агуулж болно.

1. XSD (XML Schema Definition):

XSD (XML Schema Definition) нь XML баримт бичгийн бүтцийг тодорхойлоход ашигладаг схемийн хэл юм. Энэ нь DTD (Document Type Definition)-тэй төстэй боловч илүү хүчирхэг, уян хатан байдаг. XSD нь хэрэглэгчийн тодорхойлсон нийлмэл төрлүүд болон суурилагдсан өгөгдлийн төрлийг хязгаарлах эсвэл өргөтгөх чадвар гэх мэт илүү төвөгтэй төрлийн тодорхойлолтыг зөвшөөрдөг.

XSD нь XML дээр бичигдсэн бөгөөд ихэвчлэн .xsd өргөтгөлтэй тусдаа файлд тодорхойлогддог. XML баримт бичиг нь XML жишээ баримт дахь "schemaLocation" шинж чанарыг ашиглан бүтцийг нь зааж өгөхийн тулд XSD файлыг лавлаж болно.

XML баримт бичгийн бүтцийг тодорхойлсон XSD схемийн жишээ :



Энэ жишээн дээр XSD схем нь нэг буюу хэд хэдэн "ном" элемент агуулсан "номын дэлгүүр" хэмээх үндсэн элемент бүхий XML баримт бичгийн бүтцийг тодорхойлдог. "Ном" элемент бүр "гарчиг", "зохиогч", "үнэ" элементтэй, мөн "ангилал" гэсэн нэмэлт шинж чанартай байдаг.

XSD схем нь XSD нэрийн орон зайг зарладаг "xs:schema" элементээс эхэлдэг. "xs:element" элементийг XML элементийг тодорхойлоход, "xs:complexType" элементийг XML элементийн цогц төрлийг тодорхойлоход ашигладаг. "xs:sequence" элементийг XML элемент доторх child элементүүдийн дарааллыг тодорхойлоход ашигладаг.

Ерөнхийдөө XSD нь XML баримт бичгийн нарийн төвөгтэй төрлийг тодорхойлох, баталгаажуулах боломжийг олгодог хүчирхэг, уян хатан схемийн хэл юм. Энэ нь вэб үйлчилгээ, мэдээллийн сангийн дизайн, өгөгдөл солилцох формат зэрэг олон салбар, хэрэглээнд өргөн хэрэглэгддэг.

"xs:schema" элемент нь XSD схемийн үндсэн элемент бөгөөд XSD нэрийн орон зайг зарладаг. "xmlns:xs" шинж чанар нь XSD нэрийн орон зайд нэрийн орон зайн угтварыг зааж өгдөг.

"xs:element" элементийг XML элементийг тодорхойлоход ашигладаг. Энэ тохиолдолд "номын дэлгүүр" гэж нэрлэгддэг элементийг тодорхойлдог.

"xs:complexType" элементийг XML элементийн цогц төрлийг тодорхойлоход ашигладаг. Энэ нь хүүхдийн элементүүдийн дарааллаас бүрдэх "номын дэлгүүр" элементийн цогц төрлийг тодорхойлдог.

"xs:sequence" элементийг XML элемент доторх хүүхдийн элементүүдийн дарааллыг тодорхойлоход ашигладаг. Энэ тохиолдолд олон удаа тохиолдож болох нэг "ном" элементийг агуулдаг.

"Ном" элементийг тодорхойлохын тулд "xs:element" элементийг дахин ашигладаг. Энэ нь хүүхдийн элементүүдийн дараалал ба "категори" гэж нэрлэгддэг шинж чанараас бүрдэх цогц төрөлтэй.

"Ном" элементийн хүүхэд элементүүдийг "xs:sequence" элементийг ашиглан тодорхойлсон бөгөөд "гарчиг", "зохиогч", "үнэ" элементүүдийг тус тусын төрлөөр нь агуулна.

"Xs: attribute" элемент нь "ном" элементийн шинж чанарыг тодорхойлоход хэрэглэгддэг. Энэ тохиолдолд энэ нь "xs:string" төрлийн "категори" шинж чанарыг тодорхойлдог.

Ерөнхийдөө энэхүү XSD схем нь нэг буюу хэд хэдэн "номын дэлгүүр"-ийн элементүүдийг агуулсан XML баримт бичгийн бүтцийг тодорхойлдог бөгөөд тус бүр нь "гарчиг", "зохиогч", "үнэ", "ангилал" гэсэн нэг буюу хэд хэдэн "ном" элементийг агуулдаг. "шинж чанар.

4. XSLT (Extensible Stylesheet Language Transformations)

XSLT stands for Extensible Stylesheet Language Transformations. It is a language used to transform XML documents into other formats. XSLT is designed to separate the presentation of the data from the data itself, so that the same data can be presented in different ways without changing the underlying data.

XSLT uses a set of rules to transform XML documents into other formats. These rules are defined in an XSLT stylesheet, which is a separate XML document that contains instructions for how to transform the input XML document. The rules can be simple or complex, depending on the complexity of the transformation.

XSLT can be used to transform XML documents into a variety of formats, such as HTML, PDF, or plain text. This makes it a powerful tool for publishing and sharing data across different platforms and applications.

One of the key benefits of XSLT is that it allows developers to separate the content of a document from its presentation. This makes it easier to update the presentation of a document without having to change the underlying data. It also makes it easier to reuse content across different platforms and applications.

In summary, XSLT is a language used to transform XML documents into other formats. It separates the presentation of the data from the data itself, and allows developers to define rules for how the transformation should take place. XSLT is a powerful tool for publishing and sharing data across different platforms and applications.



This is an XSLT stylesheet that transforms an XML document into an HTML table. Let's go through the steps in detail:

1. The first line declares that this is an XML document with a version number of 1.0 and an encoding of UTF-8. This is standard XML syntax.
2. The second line declares that this document is an XSLT stylesheet, and specifies the namespace for XSLT using the "xmlns" attribute. The version attribute specifies the version of XSLT that this stylesheet is written in.
3. The **<xsl:template>** element specifies the root template for the XSLT transformation. In this case, the "match" attribute is set to "/", which means that this template will match the root node of the input XML document.
4. Inside the root template, we define an HTML table with a border of 1. The table has a header row with three columns: "Title", "Author", and "Price". The background color of the header row is set to a light green color using the "bgcolor" attribute.
5. The **<xsl:for-each>** element is used to iterate over a set of nodes in the input XML document. The "select" attribute specifies the XPath expression that selects the nodes to iterate over. This XPath expression is not shown in the code you provided, but it would typically select a set of **<book>** elements in the input XML document.
6. Inside the **<xsl:for-each>** loop, we define a new row in the HTML table using the **<tr>** element. This row contains three columns, each of which contains the value of a particular element from the input XML document. The **<xsl:value-of>** element is used to output the value of a particular element in the input XML document. The "select" attribute specifies the XPath expression that selects the element to output.
7. Finally, the closing tags for the HTML table, body, and HTML elements are added to complete the HTML document.

In summary, this XSLT stylesheet takes an input XML document that contains book information and transforms it into an HTML table that displays the book titles, authors, and prices in a user-friendly way. The stylesheet uses XSLT elements and XPath expressions to extract the data from the input XML document and create an HTML table that can be displayed in a web browser.

5. XQuery

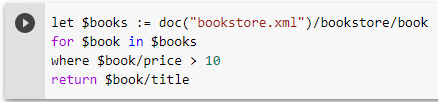
XQuery is a declarative language that is used to query and transform XML data. It is designed to be used with XML databases and allows developers to extract specific information from XML documents, manipulate the structure of the document, or combine multiple XML documents. XQuery is similar to SQL in that it allows developers to query data in a database, but XQuery is specifically designed for working with XML data.

XQuery uses a syntax that is similar to other programming languages, such as Java or C#. It includes a number of built-in functions that can be used to manipulate XML data, such as filtering, sorting, and grouping. Developers can also create their own custom functions to extend the functionality of XQuery.

One of the key features of XQuery is its ability to handle nested data structures. It can query data at any level of nesting, making it easy to extract specific information from complex XML documents. XQuery also supports the use of namespaces, which allows developers to query and manipulate XML documents that use different namespaces.

XQuery is often used in conjunction with other XML technologies such as XSLT, which is used to transform XML documents into different formats. XQuery can also be used to transform XML documents, but it is typically used for more complex transformations or for querying data from multiple sources.

Overall, XQuery is a powerful language for working with XML data. It allows developers to extract, transform, and combine data from XML documents in a flexible and efficient manner.



The XQuery code you provided is a simple example of a query that extracts data from an XML document based on a specific condition. Here is a step-by-step explanation of what the code does:

1. The first line of the code defines a variable called $books and assigns it the value of all the book elements in the XML document bookstore.xml. The doc() function is used to load the XML document.
2. The second line of the code begins a for loop that iterates over each book element in the $books variable.
3. The where clause is used to filter the books based on a condition. In this case, the condition is that the price element of the book element is greater than 10.
4. The return statement is used to specify what data should be returned for each book that meets the condition. In this case, the code returns the title element of each book.

In summary, the XQuery code loads an XML document, filters the data based on a condition, and returns a specific element of the filtered data. This is a simple example of how XQuery can be used to extract specific information from an XML document.

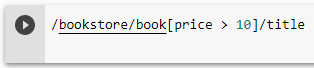
6. XPath

XPath is a language for addressing parts of an XML document, designed to be used by both XSLT and XPointer. XPath is used to navigate through elements and attributes in an XML document to locate specific data. XPath expressions are written in a syntax similar to that used in file systems, with slashes separating elements and attributes.

XPath provides a powerful set of tools for locating nodes in an XML document. Some of the key features of XPath include:

1. Path expressions: XPath uses path expressions to navigate through an XML document, much like file paths in a file system. XPath expressions can be absolute (starting at the root node) or relative (starting at the current node).
2. Selecting nodes: XPath can be used to select nodes based on their element name, attribute name, or value. For example, an XPath expression could select all book elements in an XML document with a price attribute greater than $10.
3. Predicates: Predicates are used to further refine the selection of nodes in an XPath expression. For example, an XPath expression could select all book elements with a price attribute greater than $10 and a category attribute of "fiction".
4. Functions: XPath provides a set of built-in functions that can be used to manipulate and filter data in an XML document. For example, the substring() function can be used to extract a substring of a string value.

XPath is widely used in XSLT and XQuery to locate nodes in an XML document and transform or extract data from the document. Understanding XPath is an essential part of working with XML and related technologies.



In this example, the XPath expression selects the title elements of all books in the "bookstore" document that have a price greater than 10.

Overall, these XML technologies are essential tools for working with XML data. XML is used to describe data, DTD and XSD are used to define the structure of XML documents, XSLT is used to transform XML documents, XQuery is used to query XML data, and XPath is used to navigate XML documents. By understanding these technologies and how to use them effectively, developers can create powerful applications that work with XML data.

The XPath expression **/bookstore/book[price > 10]/title** is used to select all **title** elements in an XML document that satisfy the condition **price > 10**.

Here is a step-by-step explanation of how this XPath expression works:

1. **/bookstore** - Starting at the root node, select the **bookstore** element.
2. **/bookstore/book** - Select all **book** elements that are children of the **bookstore** element.
3. **/bookstore/book[price > 10]** - Filter the selected **book** elements by selecting only those that have a **price** child element whose value is greater than 10.
4. **/bookstore/book[price > 10]/title** - For each selected **book** element, select the **title** child element.

So the result of this XPath expression is a list of all **title** elements in the XML document that belong to a **book** element with a **price** element greater than 10.

Overall, this XPath expression demonstrates the power and flexibility of XPath in selecting specific elements or attributes from an XML document based on specified criteria. It is a valuable tool for navigating and manipulating XML data.