**DOCUMENT TYPE DEFINITION**

Adopted from[**https://www.xmlfiles.com/dtd/dtd-intro/**](https://www.xmlfiles.com/dtd/dtd-intro/)

**Introduction to DTD**

The purpose of a DTD is to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements. A DTD can be declared inline in your XML document, or as an external reference.

**Internal DTD**

This is an XML document with a Document Type Definition: ([Open it in IE5](https://www.xmlfiles.com/dtd/examples/note_in_dtd.xml), and select view source)

|  |
| --- |
| <?xml version="1.0"?>  <!DOCTYPE note [  <!ELEMENT note (to,from,heading,body)>  <!ELEMENT to (#PCDATA)>  <!ELEMENT from (#PCDATA)>  <!ELEMENT heading (#PCDATA)>  <!ELEMENT body (#PCDATA)>  ]>  <note>  <to>Tove</to>  <from>Jani</from>  <heading>Reminder</heading>  <body>Don't forget me this weekend!</body>  </note> |

The DTD is interpreted like this:  
**!ELEMENT note** (in line 2) defines the element “note” as having four elements: “to,from,heading,body”.  
**!ELEMENT to** (in line 3) defines the “to” element  to be of the type “CDATA”.  
**!ELEMENT from** (in line 4) defines the “from” element to be of the type “CDATA”  
and so on…..

**External DTD**

This is the same XML document with an external DTD:  ([Open  
it in IE5](https://www.xmlfiles.com/dtd/examples/note_ex_dtd.xml), and select view source)

|  |
| --- |
| <?xml version="1.0"?>  <!DOCTYPE note SYSTEM "note.dtd">  <note>  <to>Tove</to>  <from>Jani</from>  <heading>Reminder</heading>  <body>Don't forget me this weekend!</body>  </note> |

This is a copy of the file “note.dtd” containing the Document Type Definition:

|  |
| --- |
| <?xml version="1.0"?>  <!ELEMENT note (to,from,heading,body)>  <!ELEMENT to (#PCDATA)>  <!ELEMENT from (#PCDATA)>  <!ELEMENT heading (#PCDATA)>  <!ELEMENT body (#PCDATA)> |

**Why use a DTD?**

XML provides an application independent way of sharing data. With a DTD, independent groups of people can agree to use a common DTD for interchanging data. Your application can use a standard DTD to verify that data that you receive from the outside world is valid. You can also use a DTD to verify your own data.

DTD – Elements

Declaring an Element

In the DTD, XML elements are declared with an element declaration. An element declaration has the following syntax:

|  |
| --- |
| <!ELEMENT element-name (element-content)> |

Empty elements

Empty elements are declared with the keyword EMPTY inside the parentheses:

|  |
| --- |
| <!ELEMENT element-name (EMPTY)>  example:  <!ELEMENT img (EMPTY)> |

Elements with data

Elements with data are declared with the data type inside parentheses:

|  |
| --- |
| <!ELEMENT element-name (#CDATA)>  or  <!ELEMENT element-name (#PCDATA)>  or  <!ELEMENT element-name (ANY)>  example:  <!ELEMENT note (#PCDATA)> |

#CDATA means the element contains character data that is not supposed to be parsed by a parser.  
#PCDATA means that the element contains data that IS going to be parsed by a parser.  
The keyword ANY declares an element with any content.

If a #PCDATA section contains elements, these elements must also be declared.

Elements with children (sequences)

Elements with one or more children are defined with the name of the children elements inside  
the parentheses:

|  |
| --- |
| <!ELEMENT element-name (child-element-name)>  or  <!ELEMENT element-name (child-element-name,child-element-name,.....)>  example:  <!ELEMENT note (to,from,heading,body)> |

When children are declared in a sequence separated by commas, the children must appear in the same sequence in the document. In a full declaration, the children must also be declared, and the children can also have children.  
The full declaration of the note document will be:

|  |
| --- |
| <!ELEMENT note (to,from,heading,body)>  <!ELEMENT to (#CDATA)>  <!ELEMENT from (#CDATA)>  <!ELEMENT heading (#CDATA)>  <!ELEMENT body (#CDATA)> |

Wrapping

If the DTD is to be included in your XML source file, it should be wrapped in a DOCTYPE definition with the following syntax:

|  |
| --- |
| <!DOCTYPE root-element [element-declarations]>  example:  <?xml version="1.0"?>  <!DOCTYPE note [  <!ELEMENT note (to,from,heading,body)>  <!ELEMENT to (#CDATA)>  <!ELEMENT from (#CDATA)>  <!ELEMENT heading (#CDATA)>  <!ELEMENT body (#CDATA)>  ]>  <note>  <to>Tove</to>  <from>Jani</from>  <heading>Reminder</heading>  <body>Don't forget me this weekend</body>  </note> |

Declaring only one occurrence of the same element

|  |
| --- |
| <!ELEMENT element-name (child-name)>  example  <!ELEMENT note (message)> |

The example declaration above declares that the child element message can only occur one time inside the note element.

Declaring minimum one occurrence of the same element

|  |
| --- |
| <!ELEMENT element-name (child-name+)>  example  <!ELEMENT note (message+)> |

The + sign in the example above declares that the child element message must occur one or more times inside the note element.

Declaring zero or more occurrences of the same element

|  |
| --- |
| <!ELEMENT element-name (child-name\*)>  example  <!ELEMENT note (message\*)> |

The \* sign in the example above declares that the child element message can occur zero or more times inside the note element.

Declaring zero or one occurrences of the same element

|  |
| --- |
| <!ELEMENT element-name (child-name?)>  example  <!ELEMENT note (message?)> |

The ? sign in the example above declares that the child element message can occur zero or one times inside the note element.

Declaring mixed content

|  |
| --- |
| example  <!ELEMENT note (to+,from,header,message\*,#PCDATA)> |

The example above declares that the element note must contain at least one **to** child element, exactly one **from** child element, exactly one **header**, zero or more **message**, and some other parsed **character data** as well

DTD – Attributes

Declaring Attributes

In the DTD, XML element attributes are declared with an ATTLIST declaration. An attribute declaration has the following syntax:

|  |
| --- |
| <!ATTLIST element-name attribute-name attribute-type default-value> |

As you can see from the syntax above, the ATTLIST declaration defines the element which can have the attribute, the name of the attribute, the type of the attribute, and the default attribute value.

The **attribute-type** can have the following values:

|  |  |
| --- | --- |
| **VALUE** | **EXPLANATION** |
| CDATA | The value is character data |
| (eval|eval|..) | The value must be an enumerated value |
| ID | The value is an unique id |
| IDREF | The value is the id of another element |
| IDREFS | The value is a list of other ids |
| NMTOKEN | The value is a valid XML name |
| NMTOKENS | The value is a list of valid XML names |
| ENTITY | The value is an entity |
| ENTITIES | The value is a list of entities |
| NOTATION | The value is a name of a notation |
| xml: | The value is predefined |

The **attribute-default-value** can have the following values:

|  |  |
| --- | --- |
| **VALUE** | **EXPLANATION** |
| #DEFAULT value | The attribute has a default value |
| #REQUIRED | The attribute value must be included in the element |
| #IMPLIED | The attribute does not have to be included |
| #FIXED value | The attribute value is fixed |

Attribute declaration example

|  |
| --- |
| DTD example:  <!ELEMENT square EMPTY>  <!ATTLIST square width CDATA "0">  XML example:  <square width="100"></square> |

In the above example the element square is defined to be an empty element with the attributes width of  type CDATA. The width attribute has a default value of 0.

Default attribute value

|  |
| --- |
| Syntax:  <!ATTLIST element-name attribute-name CDATA "default-value">  DTD example:  <!ATTLIST payment type CDATA "check">  XML example:  <payment type="check"> |

Specifying a default value for an attribute, assures that the attribute will get a value even if the author of the XML document didn’t include it.

Implied attribute

|  |
| --- |
| Syntax:  <!ATTLIST element-name attribute-name attribute-type #IMPLIED>  DTD example:  <!ATTLIST contact fax CDATA #IMPLIED>  XML example:  <contact fax="555-667788"> |

Use an implied attribute if you don’t want to force the author to include an attribute and you don’t have an option for a default value either.

Required attribute

|  |
| --- |
| Syntax:  <!ATTLIST element-name attribute\_name attribute-type #REQUIRED>  DTD example:  <!ATTLIST person number CDATA #REQUIRED>  XML example:  <person number="5677"> |

Use a required attribute if you don’t have an option for a default value, but still want to force the attribute to be present.

Fixed attribute value

|  |
| --- |
| Syntax:  <!ATTLIST element-name attribute-name attribute-type #FIXED "value">  DTD example:  <!ATTLIST sender company CDATA #FIXED "Microsoft">  XML example:  <sender company="Microsoft"> |

Use a fixed attribute value when you want an attribute to have a fixed value without allowing the author to change it. If an author includes another value, the XML parser will return an error.

Enumerated attribute values

|  |
| --- |
| Syntax:  <!ATTLIST element-name attribute-name (eval|eval|..) default-value>  DTD example:  <!ATTLIST payment type (check|cash) "cash">  XML example:  <payment type="check">  or  <payment type="cash"> |

Use enumerated attribute values when you want the attribute values to be one of a fixed set of legal values.

**DTD – Entities**

**Entities**

Entities as variables used to define shortcuts to common text.

Entity references are references to entities.

Entities can be declared internal.

Entities can be declared external

**Internal Entity Declaration**

|  |
| --- |
| Syntax:  <!ENTITY entity-name "entity-value">  DTD Example:  <!ENTITY writer "Jan Egil Refsnes.">  <!ENTITY copyright "Copyright XML101.">  XML example:  <author>&writer;&copyright;</author> |

**External Entity Declaration**

|  |
| --- |
| Syntax:  <!ENTITY entity-name SYSTEM "URI/URL">  DTD Example:  <!ENTITY writer SYSTEM "http://www.xml101.com/entities/entities.xml">  <!ENTITY copyright SYSTEM "http://www.xml101.com/entities/entities.dtd">  XML example:  <author>&writer;&copyright;</author> |

[DTD Introduction](https://www.xmlfiles.com/dtd/dtd-intro/)

[DTD Elements](https://www.xmlfiles.com/dtd/dtd-elements/)

[DTD Attributes](https://www.xmlfiles.com/dtd/dtd-attributes/)

[DTD Entities](https://www.xmlfiles.com/dtd/dtd-entities/)

[DTD Validation](https://www.xmlfiles.com/dtd/dtd-validation/)

[DTD Examples](https://www.xmlfiles.com/dtd/dtd-examples/)

DTD Validation

Validating with the XML Parser

If you try to open an XML document, the XML Parser might generate an error. By accessing the parseError object, the exact error code, the error text, and even the line that caused the error can be retrieved:

|  |
| --- |
| var xmlDoc = new ActiveXObject("Microsoft.XMLDOM")  xmlDoc.async="false"  xmlDoc.validateOnParse="true"  xmlDoc.load("note\_dtd\_error.xml")  document.write("<br>Error Code: ")  document.write(xmlDoc.parseError.errorCode)  document.write("<br>Error Reason: ")  document.write(xmlDoc.parseError.reason)  document.write("<br>Error Line: ")  document.write(xmlDoc.parseError.line) |

[Try it Yourself](https://www.xmlfiles.com/dtd/examples/tryit.asp?filename=note_error_three) or or just [look at the XML file](https://www.xmlfiles.com/dtd/examples/note_dtd_error.xml)

Turning Validation off

Validation can be turned off by setting the XML parser’s validateOnParse=”false”.

|  |
| --- |
| var xmlDoc = new ActiveXObject("Microsoft.XMLDOM")  xmlDoc.async="false"  xmlDoc.validateOnParse="false"  xmlDoc.load("note\_dtd\_error.xml")  document.write("<br>Error Code: ")  document.write(xmlDoc.parseError.errorCode)  document.write("<br>Error Reason: ")  document.write(xmlDoc.parseError.reason)  document.write("<br>Error Line: ")  document.write(xmlDoc.parseError.line) |

**DTD – Examples from the Net**

**TV Scedule DTD**

By David Moisan. Copied from his Web: <http://www.davidmoisan.org/>

|  |
| --- |
| <!DOCTYPE TVSCHEDULE [ <!ELEMENT TVSCHEDULE (CHANNEL+)> <!ELEMENT CHANNEL (BANNER, DAY+)> <!ELEMENT BANNER (#PCDATA)> <!ELEMENT DAY ((DATE, HOLIDAY) | (DATE, PROGRAMSLOT+))+> <!ELEMENT HOLIDAY (#PCDATA)> <!ELEMENT DATE (#PCDATA)> <!ELEMENT PROGRAMSLOT (TIME, TITLE, DESCRIPTION?)> <!ELEMENT TIME (#PCDATA)> <!ELEMENT TITLE (#PCDATA)> <!ELEMENT DESCRIPTION (#PCDATA)>  <!ATTLIST TVSCHEDULE NAME CDATA #REQUIRED> <!ATTLIST CHANNEL CHAN CDATA #REQUIRED> <!ATTLIST PROGRAMSLOT VTR CDATA #IMPLIED> <!ATTLIST TITLE RATING CDATA #IMPLIED> <!ATTLIST TITLE LANGUAGE CDATA #IMPLIED>  ]> |

**A Report DTD**

By Richard Erlander. Copied from his Web: <http://pdbeam.uwaterloo.ca/~rlander/>

|  |
| --- |
| <!DOCTYPE REPORT [  <!ELEMENT REPORT (TITLE,(SECTION|SHORTSECT)+)> <!ELEMENT SECTION (TITLE,%BODY;,SUBSECTION\*)> <!ELEMENT SUBSECTION (TITLE,%BODY;,SUBSECTION\*)> <!ELEMENT SHORTSECT (TITLE,%BODY;)> <!ELEMENT TITLE %TEXT;> <!ELEMENT PARA %TEXT;> <!ELEMENT LIST (ITEM)+> <!ELEMENT ITEM (%BLOCK;)> <!ELEMENT CODE (#PCDATA)> <!ELEMENT KEYWORD (#PCDATA)> <!ELEMENT EXAMPLE (TITLE?,%BLOCK;)> <!ELEMENT GRAPHIC EMPTY>  <!ATTLIST REPORT security (high | medium | low ) “low”> <!ATTLIST CODE type CDATA #IMPLIED> <!ATTLIST GRAPHIC file ENTITY #REQUIRED>  <!ENTITY xml “Extensible Markup Language”> <!ENTITY sgml “Standard Generalized Markup Language”> <!ENTITY pxa “Professional XML Authoring”> <!ENTITY % TEXT “(#PCDATA|CODE|KEYWORD|QUOTATION)\*”> <!ENTITY % BLOCK “(PARA|LIST)+”> <!ENTITY % BODY “(%BLOCK;|EXAMPLE|NOTE)+”>  <!NOTATION GIF SYSTEM “”> <!NOTATION JPG SYSTEM “”> <!NOTATION BMP SYSTEM “”>  ]> |

**Newspaper Article DTD**

Copied from <http://www.vervet.com/>

|  |
| --- |
| <!DOCTYPE NEWSPAPER [  <!ELEMENT NEWSPAPER (ARTICLE+)> <!ELEMENT ARTICLE (HEADLINE, BYLINE, LEAD, BODY, NOTES)> <!ELEMENT HEADLINE (#PCDATA)> <!ELEMENT BYLINE (#PCDATA)> <!ELEMENT LEAD (#PCDATA)> <!ELEMENT BODY (#PCDATA)> <!ELEMENT NOTES (#PCDATA)>  <!ATTLIST ARTICLE AUTHOR CDATA #REQUIRED> <!ATTLIST ARTICLE EDITOR CDATA #IMPLIED> <!ATTLIST ARTICLE DATE CDATA #IMPLIED> <!ATTLIST ARTICLE EDITION CDATA #IMPLIED>  <!ENTITY NEWSPAPER “Vervet Logic Times”> <!ENTITY PUBLISHER “Vervet Logic Press”> <!ENTITY COPYRIGHT “Copyright 1998 Vervet Logic Press”>  ]> |

**Product Catalog DTD**

Copied from <http://www.vervet.com/>

|  |
| --- |
| <!DOCTYPE CATALOG [  <!ELEMENT CATALOG (PRODUCT+)> <!ELEMENT PRODUCT (SPECIFICATIONS+, OPTIONS?, PRICE+, NOTES?)> <!ELEMENT SPECIFICATIONS (#PCDATA)> <!ELEMENT OPTIONS (#PCDATA)> <!ELEMENT PRICE (#PCDATA)> <!ELEMENT NOTES (#PCDATA)>  <!ATTLIST PRODUCT NAME CDATA #IMPLIED> <!ATTLIST CATEGORY (HandTool | Table | Shop-Professional) "HandTool"> <!ATTLIST PARTNUM CDATA #IMPLIED> <!ATTLIST PLANT (Pittsburgh | Milwaukee | Chicago) "Chicago"> <!ATTLIST INVENTORY (InStock | Backordered | Discontinued) "InStock"> <!ATTLIST SPECIFICATIONS WEIGHT CDATA #IMPLIED> <!ATTLIST POWER CDATA #IMPLIED> <!ATTLIST OPTIONS FINISH (Metal | Polished | Matte) "Matte"> <!ATTLIST OPTIONS ADAPTER (Included | Optional | NotApplicable) "Included"> <!ATTLIST OPTIONS CASE (HardShell | Soft | NotApplicable) "HardShell"> <!ATTLIST PRICE MSRP CDATA #IMPLIED> <!ATTLIST PRICE WHOLESALE CDATA #IMPLIED> <!ATTLIST PRICE STREET CDATA #IMPLIED> <!ATTLIST PRICE SHIPPING CDATA #IMPLIED>  <!ENTITY AUTHOR "John Doe"> <!ENTITY COMPANY "JD Power Tools, Inc."> <!ENTITY EMAIL "jd@jd-tools.com">  ]> |