

# DTD

(10주차)

# 학습개요

- 학습 목표

- DTD를 구성하는 빌딩 블록(애트리뷰트, 엔터티)에 대해 설명할 수 있다.
- DTD를 이용한 유효성 검사 방법을 설명할 수 있다.
- 유효한 문서를 작성하는 방법을 설명할 수 있다.

- 학습 내용

- DTD 빌딩 블록
- DTD 유효성 검사
- DTD 실습

# 애트리뷰트

- ID 애트리뷰트 선언

<!ATTLIST element-name attribute-name ID #REQUIRED>

DTD:

<!ATTLIST author id ID #REQUIRED>

XML:

<author id="p1001" /> <!-- 유효한 XML -->

<author id="1001" /> <!-- 유효하지 않은 XML -->

<author id="p100 1" /> <!-- 유효하지 않은 XML -->

# 애트리뷰트

- IDREF 애트리뷰트 선언

`<!ATTLIST element-name attribute-name IDREF #REQUIRED>`

DTD:

`<!ATTLIST type id ID #REQUIRED>`

`<!ATTLIST book id ID #REQUIRED  
type IDREF #REQUIRED>`

XML:

`<type id="t01" />`

`<book id="b01" type="t01" /> <!-- 유효한 XML -->`

# 애트리뷰트

- IDREFS 애트리뷰트 선언

`<!ATTLIST element-name attribute-name IDREFS #IMPLIED>`

DTD:

`<!ATTLIST type id ID #REQUIRED>`

`<!ATTLIST book id ID #REQUIRED  
types IDREFS #IMPLIED>`

XML:

`<type id="t01" />`

`<type id="t02" />`

`<book id="b01" types="t01 t02" /> <!-- 유효한 XML -->`

# 애트리뷰트

- NMTOKEN 애트리뷰트 선언

<!ATTLIST element-name attribute-name NMTOKEN #REQUIRED>

DTD:

<!ATTLIST author name NMTOKEN #REQUIRED>

XML:

<author name="홍길동" /> <!-- 유효한 XML -->

<author name="홍 길동" /> <!-- 유효하지 않은 XML -->

# 애트리뷰트

- NMTOKENS 애트리뷰트 선언

<!ATTLIST element-name attribute-name NMTOKENS #REQUIRED>

DTD:

<!ATTLIST author name NMTOKENS #REQUIRED>

XML:

<author name="홍길동 이순신 강감찬" /> <!-- 유효한 XML -->

# 엔터티

- 내부 엔터티 선언  
    <!ENTITY entity-name "entity-value">

DTD Example:

```
<!ENTITY kr "한국어">  
<!ENTITY en "영어" >
```

XML example:

```
<language>&kr;</language>  
<description language="&en;">...</language>
```



# 엔터티

- 외부 엔터티 선언  
`<!ENTITY entity-name SYSTEM "URI/URL">`

DTD Example:

```
<!ENTITY lang SYSTEM "entities.xml">
```

XML example:

```
<?xml version="1.0" standalone="no" encoding="UTF-8"?>  
<list> &lang;</list>
```

entities.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
  
<languages>  
  <language id="kr">한국어  
</language>  
  <language id="en">영어</language>  
</languages>
```

# 엔터티

- 내부 파라미터 엔터티 선언  
    <!ENTITY % entity-name "entity-value">

DTD Example:

```
<!ENTITY % contact "(name, phone, email)">
```

```
<!ENTITY % txt "(#PCDATA)">
```

```
<!ELEMENT contacts %contact;>
```

```
<!ELEMENT name %txt;>
```

```
<!ELEMENT phone %txt;>
```

```
<!ELEMENT email %txt;>
```

# 엔터티

- 외부 파라미터 엔터티 선언  
<!ENTITY % entity-name SYSTEM "URI/URL">

DTD Example:

```
<!ENTITY entities SYSTEM "entities.dtd">
```

**%entities;**

```
<!ELEMENT contacts %contact;>
```

```
<!ELEMENT name %txt;>
```

```
<!ELEMENT phone %txt;>
```

```
<!ELEMENT email %txt;>
```

entities.dtd

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<!ENTITY % contact "(name, phone, email)"
```

```
<!ENTITY % txt "(#PCDATA)">
```

# 내부 DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note [
<!ELEMENT note (#PCDATA|to|from|heading|body)*>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
  <to>홍길동</to>
  <from>이순신</from>
  <heading>공지</heading>
  <body>퇴근 후 회식!!!</body>
</note>
```

# 외부 DTD

## note.dtd

```
<?xml version="1.0" encoding="UTF-8"?>  
<!ELEMENT note (#PCDATA|to|from|heading|body)*>  
<!ELEMENT to (#PCDATA)>  
<!ELEMENT from (#PCDATA)>  
<!ELEMENT heading (#PCDATA)>  
<!ELEMENT body (#PCDATA)>
```

## note.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>  
<!DOCTYPE note SYSTEM "note.dtd">  
<note>  
  <to>홍길동</to>  
  <from>이순신</from>  
  <heading>공지</heading>  
  <body>퇴근 후 회식!!!</body>  
</note>
```

# PUBLIC 외부 DTD

- DTD가 외부에서 사용될 경우 FPI(Formal Public Identifier) 표준 방식의 DTD명을 사용해야 함
- 표준이 아닐 경우  
PUBLIC "-//owner//DTD description//XX  
  
PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
- 표준일 경우  
PUBLIC "-//owner//DTD description//XX

# PUBLIC 외부 DTD

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>  
<!DOCTYPE note PUBLIC "-//Example, Inc//DTD NoteML  
1.0//EN" "http://www.example.com/dtd/note.dtd">  
<note>  
  <to>홍길동</to>  
  <from>이순신</from>  
  <heading>공지</heading>  
  <body>퇴근 후 회식!!!</body>  
</note>
```

# DTD의 장단점

- 장점

- 내부 DTD를 사용할 수 있어 빠른 개발이 가능
- 엔터티를 사용자 정의해 사용 가능
- 대부분의 XML 파서들이 지원


- 단점

- XML 구문을 사용하지 않음
- 네임스페이스를 지원하지 않음
- 정수, 날짜 등의 데이터 타입을 지원하지 않음



# 온라인 유효성 검사

- <http://validator.w3.org/>

 **Markup Validation Service**  
Check the markup (HTML, XHTML, ...) of Web documents

Validate by URI

Validate by File Upload

Validate by Direct Input

Validate by URI


Validate a document online:


Address:

▸ More Options

Check

This validator checks the [markup validity](#) of Web documents in HTML, XHTML, SMIL, MathML, etc. If you wish to validate specific content such as [RSS/Atom feeds](#) or [CSS stylesheets](#), [MobileOK content](#), or to [find broken links](#), there are [other validators and tools](#) available. As an alternative you can also try our [non-DTD-based validator](#).

 [Try now the W3C Validator Suite™](#) premium service that checks your entire website and evaluates its conformance with W3C open standards to quickly identify those portions of your website that need your attention.



The W3C validators are developed with assistance from the Mozilla Foundation, and supported by community donations. [Donate](#) and help us build better tools for a better web.

5252

Flattr

[Home](#) [About...](#) [News](#) [Docs](#) [Help & FAQ](#) [Feedback](#) [Contribute](#)

# 학습정리

- DTD는 엘리먼트, 애트리뷰트, 엔터티를 이용해 XML 문서의 스키마(Schema)를 정의한다.
- 내부 DTD와 외부 DTD, PUBLIC 외부 DTD로 DTD를 생성할 수 있다.
- DTD를 이용한 XML 문서의 스키마에 정의되어 있는 규칙을 준수함으로써 일관성 있는 유효한 XML 문서를 생성할 수 있다.
- XML 문서를 이용한 데이터 교환 시 DTD를 사용함으로써 유효한 문서와 데이터를 확인할 수 있다.