

모바일 소프트웨어 프로젝트

XML #2. Well-formed XML



2012.03.15.

오 병 우

컴퓨터공학과

XML

언어

◆ 문법 (Syntax)

- 규칙

◆ 의미 (Semantic)

- 어순

XML

◆ Well-formed XML

- XML 언어에 맞춰서 작성한 문서

◆ Valid XML

- XML 언어에 맞추고 특정한 스키마에 맞도록 작성한 문서
- 예제
 - 이력서 양식이 많지만 지원 기관의 양식(스키마)에 맞춰서 제출
 - » 학점을 100점 만점으로 표기 등
 - 제안서 양식 등

Well-formed XML & Valid XML

Well-formed XML

- ◆ XML Specification 준수
- ◆ Well-formed가 아니면 XML이 아님

Valid XML

- ◆ XML이지만 Valid는 아닐 수 있음
- ◆ Valid XML은 Well-formed XML에 추가 조건 필요
 - DTD 또는 XML Schema 정의
 - 정의된 구조와 element 사용
- ◆ Document Type Definition(DTD)에 맞게 기술된 문서
- ◆ XML Schema에 맞게 기술된 문서

XML Grammar

- W3C(World Wide Web Consortium)의 Specification

- ◆ <http://www.w3.org/TR/2004/REC-xml-20040204>

- Specification의 문법 정의

- ◆ EBNF (Extended Backus-Naur Form) 표기법을 따름

- E.g., bookinfo := title author+ press cost?

- ◆ Expression 기호의 의미

- A?: zero or one

- A+: one or more

- A*: zero or more

- A: only one

- A B: A 뒤에 B

- A|B: A or B

Well-formed 문서

- Parser가 인식하려면 Well-formed 이어야 함
- Well-formed XML

```
[1]  document ::= prolog element Misc*
[22] prolog   ::= XMLDecl? Misc* (doctypedecl Misc*)?
[39] element ::= EmptyElemTag | STag content ETag
[27] Misc    ::= Comment | PI | S
```

- ◆ 한 개 이상의 element로 구성
- ◆ 오직 한 개의 root element
 - 가장 바깥을 root라고 하는데 같은 이름이 두 번 나오면 안됨
 - 안됨: <a> <a>
- ◆ 6개의 구성요소만 사용 가능
 - Element
 - Processing Instruction
 - DTD declaration
 - Entity
 - Comment
 - CDATA section

Well-formed XML 실습

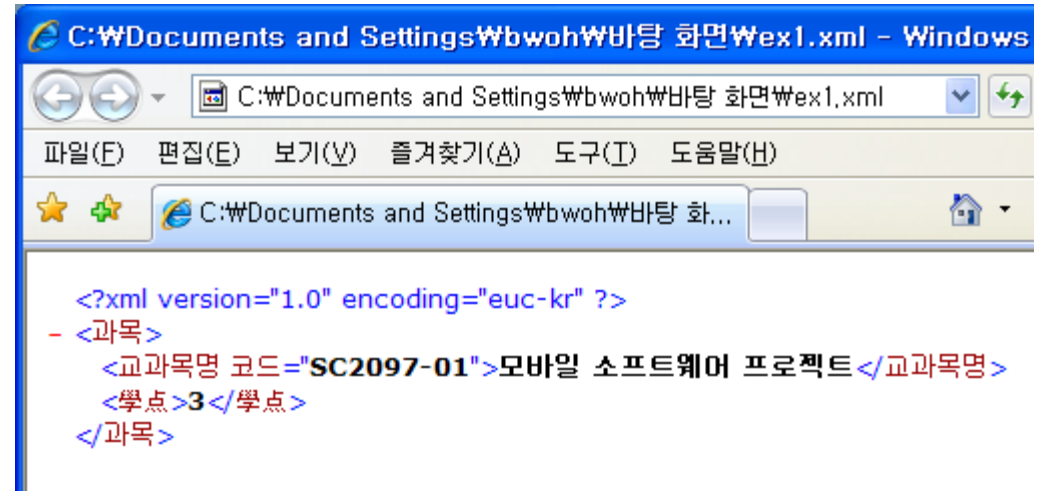
Example of Well-formed XML

Ex1.xml

생략하면
유니코드
UTF-16

```
<?xml version="1.0" encoding="euc-kr"?>
<과목>
  <교과목명 코드="SC2097-01">
    모바일 소프트웨어 프로젝트
  </교과목명>
  <學点>
    3
  </學点>
</과목>
```

Internet Explorer



Valid or Invalid?

◆ Schema가 없다면 알 수 없음

Prolog

XML 문서의 구성

```
[1] document ::= prolog element Misc*
```

Prolog의 구성

```
[22] prolog ::= XMLDecl? Misc* (doctypedecl Misc*)?
[23] XMLDecl ::= '<?xml' VersionInfo EncodingDecl? SDDDecl? S? '?>'
[24] VersionInfo ::= S 'version' Eq ('"' VersionNum '"' | "'" VersionNum "'")
[25] Eq ::= S? '=' S?
[26] VersionNum ::= '1.0'
[27] Misc ::= Comment | PI | S
[3] S ::= (#x20 | #x9 | #xD | #xA)+
```

White
Space

Processing Instructions (PI)

- XML에서 전달하는 기본 정보에는 속하지 않음

- ◆ 파서 또는 응용 프로그램에게 처리 방법 기술
- ◆ E.g., 표현을 위한 stylesheet 지정

```
[16] PI ::= '<?' 처리방법 '?>'
```

- ◆ 예제

- <?xml-stylesheet type="text/xsl" href="table.xsl"?>
- <?user-info bloodtype="A"?>

XML 선언

XML 문서의 첫번째 줄에 XML 선언

```
[1] document ::= prolog element Misc*
[22] prolog ::= XMLDecl? Misc* (doctypedecl Misc*)?
[23] XMLDecl ::= '<?xml' VersionInfo EncodingDecl? SDDDecl? S? '?>'

[24] VersionInfo ::= S 'version' Eq ("" VersionNum "" | "" VersionNum "")
[80] EncodingDecl ::= S 'encoding' Eq ("" EncName "" | "" EncName "" )
[32] SDDDecl ::= S 'standalone' Eq ("" ('yes' | 'no') "" | ("" ('yes' | 'no') ""))

[26] VersionNum ::= '1.0'
[81] EncName ::= [A-Za-z] ([A-Za-z0-9._] | '-')*
```

◆ 예제

- <?xml version="1.0" encoding="euc-kr"?>

◆ SDDDecl

- Standalone 선언
- 외부 파일 쓰는지 여부
- Default는 no (외부 파일 사용)

한글

Document Type Declarations

- DTD라고 줄여서 쓰지 않음

- Document Type 선언

- ◆ 어떤 DTD(Document Type **Definition**)를 사용하는지 지정

- DTD를 XML 문서내에 포함 가능

```
[1] document ::= prolog element Misc*
[22] prolog ::= XMLDecl? Misc* (doctypeddecl Misc*)?
[28] doctypeddecl ::= '<!DOCTYPE' S Name (S ExternalID)? S? ('[' intSubset ']' S?)? '>'

[75] ExternalID ::= 'SYSTEM' S SystemLiteral | 'PUBLIC' S PubidLiteral S SystemLiteral
[28b] intSubset ::= (markupdecl | DeclSep)*
[28a] DeclSep ::= PReference | S
[29] markupdecl ::= elementdecl | AttlistDecl | EntityDecl | NotationDecl | PI | Comment
```

- ◆ SYSTEM 사용 예제 (외부 파일 참조)

- <!DOCTYTPE 멤버 SYSTEM “member.dtd”>

- <!DOCTYTPE 멤버 SYSTEM “file:///C:/member.dtd”>

- <!DOCTYTPE 멤버 SYSTEM “http://xml.kumoh.ac.kr/ex/member.dtd”>

Document Type 선언내에 DTD 기술

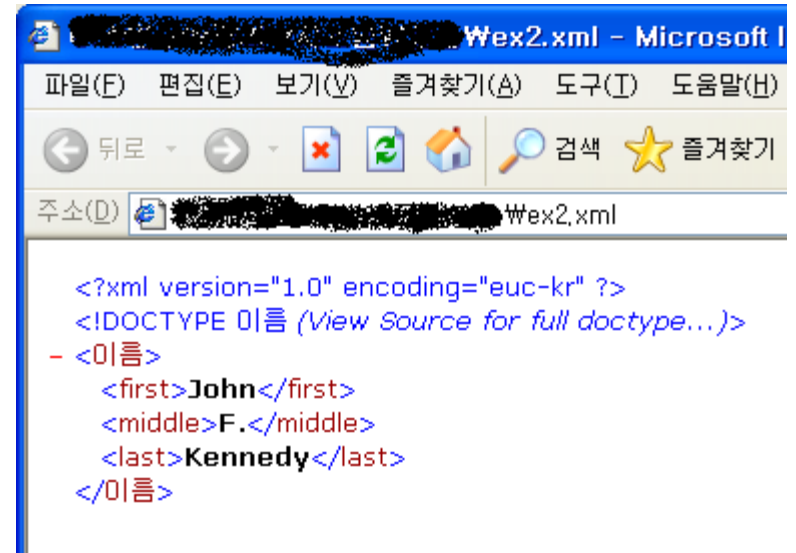
XML 문서에 DTD 포함된 예제

```
<?xml version="1.0" encoding="euc-kr" ?>
<!DOCTYPE 이름 [
  <!ELEMENT 이름 (first, middle, last)>

  <!ELEMENT first (#PCDATA)>
  <!ELEMENT middle (#PCDATA)>
  <!ELEMENT last (#PCDATA)>
]>

<이름>
  <first>John</first>
  <middle>F.</middle>
  <last>Kennedy</last>
</이름>
```

Valid한 XML 문서



Element

● 최소 구성 단위

◆ Element = Tag + Attribute + Content

```
[39] element ::= EmptyElemTag | STag content Etag

[44] EmptyElemTag ::= '<' Name (S Attribute)* S? '/>'
[40] STag ::= '<' Name (S Attribute)* S? '>' [WFC: Unique Att Spec]
[43] content ::= CharData? ((element | Reference | CDsect | PI | Comment) CharData?)*
[42] ETag ::= '</' Name S? '>'

[41] Attribute ::= Name Eq AttValue
```

◆ 예제

- <교과목명>모바일 소프트웨어 프로젝트</교과목명>
-

- <과목><교과목명>모바일 소프트웨어 프로젝트</교과목명></과목>

● 정보를 계층 구조로 표현

Tag

- 태그는 시작태그, 종료태그, 빈태그로 나누어진다.
 - ◆ 시작태그: <태그명>
 - ◆ 종료태그: </태그명>
 - ◆ 빈태그: <태그명/> = <태그명></태그명>
- 주의점
 - ◆ 태그명은 대소문자 구별 (알파벳에만 해당)
 - <Title>제목</title> : 에러
 - ◆ 공백은 태그의 시작과 중간에 올 수 없다. (끝의 공백은 무시)
 - <주민등록 번호>xxxxxxx-xxxxxxx</주민등록 번호> : 에러
 - < 주민번호>xxxxxxx-xxxxxxx</ 주민번호> : 에러
 - ◆ 첫 글자로 문자와 '_', ':' 등이 올 수 있고 숫자는 올 수 없다.
 - <63빌딩> xxxxxxxx </63빌딩> : 에러
 - ◆ 태그명의 길이에는 제한이 없으나 각 파서별로 실제적인 제한이 있을 수 있다.
 - ◆ 'XML'은 대소문자에 상관없이 태그명으로 사용할 수 없다.
 - ◆ 태그는 겹쳐서 (overlapping) 사용할 수 없다.
 - <a><c></c> : 에러
 - ◆ 시작태그가 있으면 반드시 종료태그가 있어야 한다.

Attribute

- element의 특성을 나타내기 위해 사용되는 name-value pair (NV-pair)
- 사용 형식
 - ◆ Attribute name = “Attribute value”
 - ◆ 따옴표 또는 쌍따옴표 사용
 - <과목 코드=”SC1097-01” 인원=’40’>
 - 과목 element
 - » 코드 attribute
 - » 인원 attribute

Content

- Element에서 전달하려는 기본 정보
 - ◆ Cf.) 추가정보: 태그
- 시작태그와 종료태그 사이에 위치
 - 자식요소
 - `<book><bookauthor>James Clark</bookauthor></book>`
 - 문자 데이터가 온다. 최하위 요소 (leaf node)
 - `<bookauthor>James Clark</bookauthor>`
 - Mixed
 - `<book>Easy XML<bookauthor>James Clark</bookauthor></book>`
 - CDATA section 사이에 문자 데이터가 온다
 - `<equation><![CDATA[10 < x < 15]]></equation>`
- Whitespace
 - ◆ IE로 볼 때는 무시되는 것 같지만 이는 HTML 형식으로 변환해서 보여주기 때문임
 - ◆ 응용 프로그램에서 사용시는 whitespace 보존
- 특수기호는 사용 불가
 - ◆ Entity 또는 CDATA section으로 처리 가능

Reference

Entity Reference

- ◆ 미리 내용을 정의하고 반복 사용
- ◆ 재사용 가능한 개체
 - 5개의 Predefined Entities
 - XML 표준에서 정의
 - User Defined Entity (매우 복잡)
 - DTD에서 선언 (General Entity)
 - » <!ENTITY copyright "Kumoh National Institute of Technology">
 - XML에서 사용
 - » ©right;

내장개체	표현
<	<
>	>
&	&
”	"
'	'

Character Reference

- ◆ 문자의 10진수 또는 16진수로 참조
 - 예제: ᄀ
 - ᄀ

16진수

Comments

주석

- ◆ <!-- 와 --> 사이에 '--'를 제외한 모든 문자
- ◆ 파서에서의 처리
 - 응용프로그램에 전달할 의무 없음

주의

- ◆ 첫 줄에는 안됨
- ◆ Tag 안에는 안됨
- ◆ Nested Comments 안됨

CDATA Sections

- 파싱되지 않는 문자 데이터 영역
 - ◆ “<![CDATA[” 와 “]]>” 사이
 - ◆ 파서로부터 특수 문자들을 보호하기 위해 사용
 - <![CDATA[You&Me]]>