3. XML 기본 문법

XML 문서

- **정형화된 XML 문서 (Well-Formed XML Documents)**
  - XML 권고안에 정의된 XML 문서 생성 규칙(Document Production Rules)을 잘 지켜서 작성된 문서
    - 하나의 루트 엘리먼트(root element) 존재
    - 각 엘리먼트는 시작 태그(start tag)와 종료 태그(end tag)를 가짐
      - <student> … </student>
      - <student></student> → <student />로 표기 가능
    - 엘리먼트들은 올바르게 중첩되어야 함(nested properly)

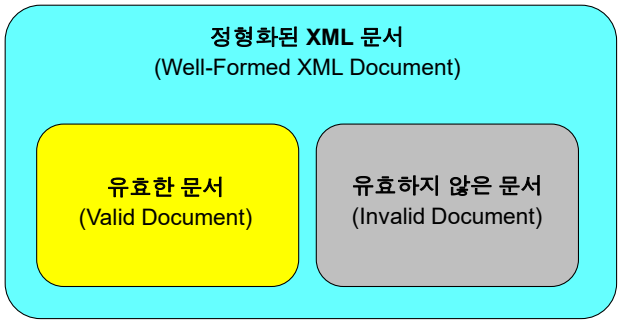
```
<student> …
  <name> …
  </name> …
</student>          (OK)
```

```
<student> …
  <name> …
</student> …
  </name>          (오류!)
```

- **유효한 XML 문서 (Valid XML Documents)**
  - 정형화된 문서이면서, XML로 정의된 특정 마크업 언어로 작성된 문서
    - DTD나 XML Schema를 통해 정의된 마크업 언어의 문법을 따르는 문서

XML 문서

- **XML 문서의 분류**



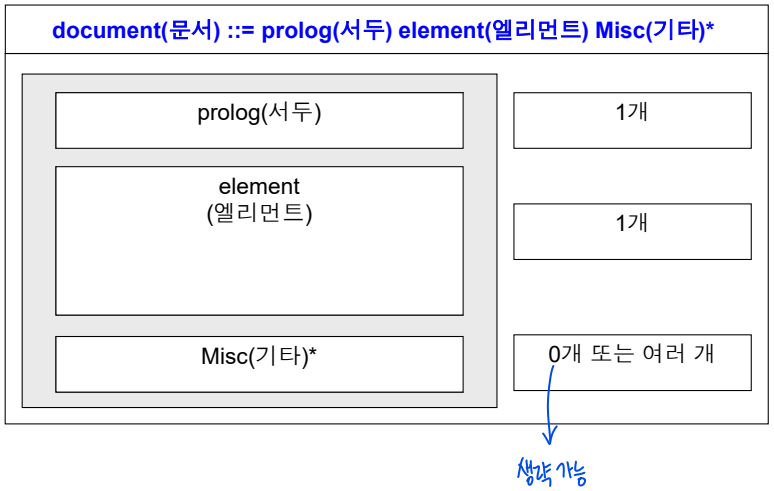
XML 문서 구조

- **XML Document Production Rules** → 없어도 있어도 될 수도, 생략될 수도 있다!

```
[1]   document  ::=  prolog element Misc*
[22]  prolog    ::=  XMLDecl? Misc* (doctypedecl Misc*)?
[23]  XMLDecl   ::=  '<?xml' VersionInfo EncodingDecl? SDDecl? S? '?>'
[27]  Misc      ::=  Comment | PI | S
[39]  element   ::=  EmptyElemTag | STag content ETag [WFC: Element Type
                     Match] [VC: Element Valid]
[43]  content   ::=  CharData? ((element | Reference | CDSect | PI | Comment)
                     CharData?)*
…                                  WFC: Well-Formedness Constraint, VC: Validity Constraint
```

  - [1]: XML 문서의 구조는 서두(prolog)와 (루트) 엘리먼트, 그리고 기타(Misc) 부분으로 구성됨
  - [22]: 서두는 XML 선언(XMLDecl), 문서 유형 선언(doctypedecl), 기타 부분으로 구성됨
  - [27]: 기타 부분은 주석(Comment), PI(Processing Instruction), 공백(S)으로 구성됨
  - [39][43]: 엘리먼트는 content를 가질 수 있음, content는 문자데이터(CharData), 자식 엘리먼트, 참조(Reference), CDATA Section, PI, 주석 등을 포함할 수 있음

XML 문서 구조



XML 문서 구조

XML 문서

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

<!-- 문서 유형 선언 -->
<!DOCTYPE booklist SYSTEM "bml.dtd">

<!-- 문서의 구조를 xhtml 문서로 변경 (PI) -->
<?xml-stylesheet type="text/xsl" href="bml.xsl"?>

<booklist>
  <book id="b1" kind="k2">
    <title>XML 기초서</title>
    <author>신민철</author>
    <publisher>프리렉</publisher>
  </book>
  <book id="b2" kind="k1">
    <title>가을엔 사랑을 느끼세요</title>
    <author>이사랑</author>
    <publisher>가을문화사</publisher>
  </book>
</booklist>
```

prolog

element

XML 선언부

- **XML 선언(declaration)**

생략 가능 / 보통 생략

```
<?xml version="버전번호" encoding="인코딩방식" standalone="yes/no"?>
```

  - 현재 작성중인 문서가 XML 문서임을 명시적으로 표현하는 것
  - XML 선언은 반드시 XML 문서의 첫 줄에 나타나야 함
  - XML 선언의 시작은 "<?xml" 로 시작하며 '<?'와 'xml' 문자열 사이에 공백이 있으면 안 됨
  - 잘못된 선언 예:

```
<!-- XML 문서 선언 (주석) -->
<?xml version="1.0" encoding="UTF-8"?>
…
```

첫 번째 줄에 주석 써서 오류!

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

? 다음 띄어쓰기 → 오류!

XML 선언부

- **XML 선언에서 사용되는 속성**

(1) version

  - version 속성은 반드시 명시해야 함

```
<?xml version="1.0"?> 또는 <?xml version='1.0'?>
```

(2) encoding

  - 이 XML 문서가 어떤 인코딩 방식을 사용하는지를 지정
  - default: UTF-8 (unicode)

```
<?xml version="1.0" encoding="UTF-8"?>    (생략 가능)
<?xml version="1.0" encoding="EUC-KR"?>
```

(3) standalone

  - 외부에 정의된 DTD 문서의 참조 필요 여부를 표시
  - default: no 보통 먼저 작성해서

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

XML 선언부

- **XML 문서 인코딩**
  - XML 권고안은 XML 문서를 유니코드 UTF-8 방식으로 저장하는 것을 기본으로 하고 있음 (default)
    - HTML5, CSS, JavaScript, PHP, MySQL 등에서 사용되는 기본 encoding 방법
  - XML 문서 생성, 저장, 출력 시 동일한 인코딩 방법을 적용해야 문자들이 올바르게 해석되고 사용될 수 있음
    - 주의: 한글 Windows 운영체제에서는 EUC-KR을 확장한 완성형 한글 코드(CP949 또는 MS949)를 기본 인코딩 방법으로 사용함
    - XML 문서를 웹을 통해 공유하기 위해서는 UTF-8을 사용하는 것이 바람직함

참고: Character Encoding

- **인코딩(Encoding)**
  - XML 문서는 텍스트 형식의 문서이기 때문에 여러 가지 언어로 작성될 수 있음
  - **문자 코드(Character Code)**: 문자들의 집합과 이 문자들을 나타내기 위한 숫자들을 1:1로 정의한 것
    - 예: ASCII, EUC-KR, Unicode 등
  - **인코딩**: 특정 문자 코드를 사용하여 문자를 표현하는 것을 말함

- **KS C 5601 & EUC-KR**
  - **KS C 5601**: 한국표준협회가 제정한 한국공업표준(KS) 정보처리분야(C)의 5601번 표준안으로, 2 bytes를 사용해서 완성형 한글 문자를 표현하는 방법을 기술함
  - KS C 5636: 영문자에 대한 KS 표준안으로, 기존 ASCII 코드와 동일하나 \를 ₩로 대체
  - **EUC-KR**: Bell Lab에서 개발한 확장 유닉스 코드(**E**xtended **U**NIX **C**ode)의 변형으로, 영어는 KS C 5636을, 한글은 KS C 5601을 사용하는 것을 말함
    - 즉, 영문자는 1 byte로 표현하고 한글 문자는 2 bytes로 표현

참고: Character Encoding

- **유니코드(Unicode)**
  - 여러 언어의 인코딩 체계를 모두 포함할 수 있도록 고안된 문자 집합
  - 유니코드 인코딩 방식을 사용하면 하나의 문서를 다양한 언어로 작성할 수 있음
  - 유니코드 평면: 유니코드는 0번~16번까지 모두 17개의 평면으로 나뉘며, 각 평면은 65536(2^{16})개의 코드로 구성됨

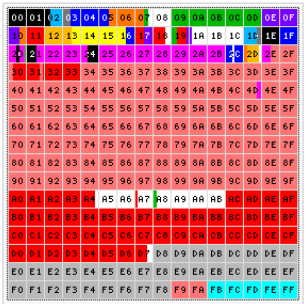
유니코드 목록 (범위)				
기본 다국어 평면	보조 다국어 평면	보조 상형 문자 평면	3차 상형 문자 평면	보조 특수 목적 평면
BMP	SMP	SIP	TIP	SSP
0000~0FFF	8000~8FFF	10000~10FFF	18000~18FFF	20000~20FFF	28000~28FFF		E0000~E0FFF
1000~1FFF	9000~9FFF	11000~11FFF	19000~19FFF	21000~21FFF	29000~29FFF		
2000~2FFF	A000~AFFF	12000~12FFF	1A000~1AFFF	22000~22FFF	2A000~2AFFF		
3000~3FFF	B000~BFFF	13000~13FFF	1B000~1BFFF	23000~23FFF	2B000~2BFFF		
4000~4FFF	C000~CFFF	14000~14FFF	1C000~1CFFF	24000~24FFF	2C000~2CFFF	문자 없음	
5000~5FFF	D000~DFFF	15000~15FFF	1D000~1DFFF	25000~25FFF	2D000~2DFFF		
6000~6FFF	E000~EFFF	16000~16FFF	1E000~1EFFF	26000~26FFF	2E000~2EFFF		
7000~7FFF	F000~FFFF	17000~17FFF	1F000~1FFFF	27000~27FFF	2F000~2FFFF		

참고: Character Encoding

- 기본 다국어 평면(BMP)
  - 유니코드의 첫번째(0번) 평면으로, U+0000부터 U+FFFF까지의 영역을 차지함. 거의 모든 근대 문자와 특수 문자가 포함되어 있으며, 그 중 대부분은 한중일 통합 한자와 한글임 (U+AC00(가) ~ U+D7A3(힣))

완성형

마지막 글자 3형



유니코드 기본 다국어 평면을 그림으로 나타낸 것이다. 각 칸은 문자 256개를 나타낸다.

- 검정: 로마자, 로마자권에서 쓰이는 기호
- 밝은 파랑: 언어학에 쓰이는 문자
- 파랑: 기타 유럽 문자
- 주황: 중동·서남 아시아에서 쓰이는 문자
- 연주황: 아프리카에서 쓰이는 문자
- 초록: 남아시아에서 쓰이는 문자
- 보라: 동남 아시아에서 쓰이는 문자
- 빨강: **동아시아에서 쓰이는 문자**
- 밝은 빨강: **한중일 통합 한자**
- 노랑: 통합 캐나다 원주민 소리 마디
- 자홍: 기호
- 어두운 회색: 발음 구별 기호
- 밝은 회색: UTF-16에서 쓰이는 상위·하위 대체 영역과 사용자 정의 영역
- 옥색: 기타 문자
- 흰색 : 쓰이지 않음

참고: Character Encoding

- **UTF-16** (Unicode Transformation Format 16-bit)
  - 기본다국어평면은 그대로 2byte 값으로 인코딩함
  - 나머지 평면들은 4byte 값으로 인코딩
  - 영어 이외의 문자로 구성된 문서의 경우 UTF-8보다 크기를 줄일 수 있음
- **UTF-8**
  - 가변길이 인코딩 방식(1byte~4byte)
  - 기본다국어평면에서 영어는 기존 ASCII 문자 코드를 그대로 사용(1byte)하고 나머지 문자들은 2~3byte로 인코딩함 (한중일 문자: 3byte)

코드 범위 (십육진법)	UTF-16BE 표현 (이진법)	UTF-8 표현 (이진법)	설명
000000-00007F	00000000 0xxxxxxx	0xxxxxxx	ASCII와 동일한 범위
000080-0007FF	00000xxx xxxxxxxx	110xxxxx 10xxxxxx	첫 바이트는 110 또는 1110으로 시작하고, 나머지 바이트들은 10으로 시작함
000800-00FFFF	xxxxxxxx xxxxxxxx	1110xxxx 10xxxxxx 10xxxxxx	
010000-10FFFF	110110yy yyxxxxxx 110111xx xxxxxxxx	11110zzz 10zzxxxx 10xxxxxx 10xxxxxx	UTF-16 서러게이트 쌍 영역 (yyyy = zzzzz - 1). UTF-8로 표시된 비트 패턴은 실제 코드 포인트와 동일하다.

XML Element

- 기본 규칙
  - 모든 XML 문서는 단 하나의 루트 엘리먼트(root element)를 가짐
  - 엘리먼트는 시작 태그와 끝 태그 한 쌍으로 구성되며 태그명은 동일해야 함
  - 시작태그와 끝 태그 사이에는 엘리먼트의 내용(content)으로 문자 데이터나 자식 엘리먼트 등이 올 수 있음
  - 엘리먼트는 부가적인 정보를 나타내는 속성(attribute)을 가질 수 있음

XML Element

<element_name> element content </element_name>

시작 태그 — 종료 태그

```
<PLAY name="Matrix" year='2000'>
  <ACT>
    <SCENE id="000">
      <TITLE>Intro</TITLE>
    </SCENE>
  </ACT>
  <POSTER image="matrix.gif"/>
</PLAY>
```

시작 태그 → <ACT>

내용(content) → <TITLE>Intro</TITLE>

종료 태그 → </ACT>

XML Element

- 엘리먼트의 종류

(1) 내용(content)을 갖는 엘리먼트

- 문자 데이터나 자식 엘리먼트를 갖는 엘리먼트

```
<book>
   <title>자연과 인간</title>
</book>
```

(2) 내용이 없는 빈 엘리먼트 (Empty Element)

- 문자 데이터나 자식 엘리먼트를 갖지 않는 엘리먼트

```
<image src="C:\temp\image1.gif"/>
→ <image src="C:\temp\image1.gif"></image> 와 동일
```

XML Element

- 엘리먼트 작성시 주의 사항

엘리먼트 작성시 주의할 점	
1	시작태그와 끝태그는 반드시 짝을 이루어야 한다. 단 내용이 없는 빈 엘리먼트는 시작 태그의 끝에 '/'를 붙여주어 시작태그인 동시에 끝태그임을 표시한다.
2	속성은 반드시 속성명=속성값 형태로 사용해야 하며 속성값은 반드시 "" 또는 ''로 감싸야 한다. 한 엘리먼트에 같은 속성명은 두 개 이상 올 수 없다.
3	태그를 나타내는 '<' 문자는 엘리먼트 내용(content)인 문자 데이터 및 속성값으로 사용할 수 없다. '>' 문자는 사용해도 되지만 가급적 사용하지 않는다.
4	시작태그의 '<' 다음에 공백문자가 올 수 없으며, 시작태그와 끝태그의 이름이 같아야 한다.
5	엘리먼트들은 올바르게 중첩되어야 한다.
6	태그 이름은 반드시 **XML 이름 작성 규칙**을 따라야 한다.

XML Element

XML 이름 작성 규칙	
1	이름은 문자(한글도 포함)나 "_"로 시작할 수 있으나 숫자나 '.' 등으로 시작할 수 없다.
2	두 번째 문자부터는 숫자 및 "_", "-", "." 등도 가능하다.
3	태그 이름은 공백문자를 포함할 수 없다.
4	':' 문자는 쓸 수는 있지만 네임스페이스와 관련된 기호이므로 사용하지 않는 것이 좋다
5	태그 이름은 대소문자를 구별한다.
6	태그 이름은 'xml'이나 'XML'로 시작할 수 없다.

< 올바른 태그 이름 >

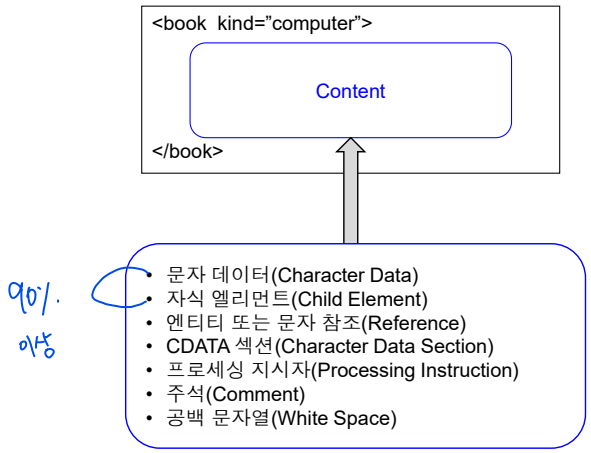
```
<book>
<_book>
<책>
<book1>
<book-1>
<Book>
```

< 잘못된 태그 이름 >

<7Book>	첫 글자는 숫자를 사용할 수 없다.
<c++>	'_', '-', '.', ':' 이외의 특수 문자는 사용할 수 없다.
<book list>	태그 이름에 공백을 사용할 수 없다.
< book>	'<' 다음에 공백을 두어서는 안 된다.
<xml-book>	태그 이름이 xml로 시작하면 안 된다.

XML Element

- 엘리먼트의 내용(Content)으로 올 수 있는 것들



XML Element

- 자식 엘리먼트
  - 각 엘리먼트의 내용으로 자식 엘리먼트(child element)를 포함

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<booklist>
    <book>
        <title> XML And VB </title>
        <publisher>프리렉</publisher>
    </book>
</booklist>
```

✓ book은 booklist의 자식 엘리먼트 ↔ booklist는 book의 부모 엘리먼트

✓ title과 publisher는 book의 자식 엘리먼트

문자 데이터

- 문자 데이터(**Character Data**)
  - XML 프로세서가 해석할 수 있는 내용 중에서 마크업을 제외한 부분

XML markups의 경우

XML 선언	<?xml version="1.0" encoding="UTF-8"?>
문서 유형 선언	<!DOCTYPE booklist SYSTEM "bml.dtd">
프로세싱 지시자(PI)	<?xml-stylesheet type="text/xml" href="bml.xsl"?>
주석	<!-- 주석 내용 -->
시작태그 및 끝태그	<book> </book>
빈 엘리먼트 태그	<imgae src="image1.gif"/>
엔티티 참조	DTD에 정의되어 있는 엔티티 참조 (예) &pub1;
문자 참조	&#10진수; &#x16진수;
CDATA 섹션 구분자	<![CDATA[문자 데이터]]>
최상위 공백 문자열	루트 엘리먼트 외부에 있는 공백 문자열
Text 선언	<?xml version="1.0" encoding="UTF-8"?>

문자 데이터

- 문자 데이터 내에는 '&' 문자와 '<' 문자를 사용할 수 없음
  - '&' 문자는 엔티티(entity) 참조의 시작을 의미
  - '<' 문자는 엘리먼트의 태그, 또는 CDATA 섹션의 시작을 의미
- **XML 문서에서 특수문자를 표현하는 방법**
  - 개체 참조(Entity Reference)
  - 문자 참조(Character Reference)

개체 참조

- 미리 정의된(**built-in**) 개체 참조명

표현 문자	문자 코드	개체 참조명	어 원
<	<	<	less then
>	>	>	greater then
'	'	'	apostrophe
"	"	"	quotation marks
&	&	&	ampersand

- 사용 예

```
<students>
  <student>
    <sid>100</sid>
    <name>홍길동</name>
    <study>&lt; 과목 &gt; 웹서비스</study>
  </student>
</students>
```

<study><과목> 웹서비스</study>

문자 참조

- 문자 참조
  - 문자 코드(Unicode) 값을 직접 사용하여 문자를 나타냄
  - 여러 가지 특수기호나 특수문자를 사용해야 할 때 유용

&#10진수 문자코드; → 10진수로 문자코드를 지정
&#x16진수 문자코드; → 16진수로 문자코드를 지정

- 사용 예

```
<students>
  <student>
    <sid>100</sid>
    <name>홍길동</name>
    <telephone> &#x260F; 02-123-6399</telephone>
    <age> &#x2661; 30 </age>
  </student>
</students>
```

☏ 02-123-6399
♡ 30

CDATA Section

- 대부분의 문자 데이터인 **PCDATA**(**P**arsed **C**haracter **DATA**)는 XML 파서가 해석(parsing)하는 데이터를 말함
- **CDATA 섹션(Section)** 내에 정의된 문자 데이터는 XML 파서가 해석하지 않고 바로 응용 프로그램(application)에게 전달
- 특수기호가 많은 경우 CDATA 섹션을 사용하면 편리

```
<?xml version="1.0" encoding="UTF-8"?>
<Root>
if (a > 0 && a < 3)  ←— 오류 발생!
</Root>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<Root>
if (a &gt; 0 &amp;&amp; a &lt; 3)
</Root>
```

⇨

```
<?xml version="1.0" encoding="UTF-8"?>
<Root>
<![CDATA[ if (a > 0 && a < 3) ]]>
</Root>
```


CDATA Section

- 주의 사항
  - '<![CDATA[' 사이나 ']]>' 사이에 공백을 둘 수 없음
  - CDATA 섹션 안에 다른 CDATA 섹션을 포함할 수 없음
  - 키워드 CDATA는 반드시 대문자를 사용함
  - CDATA 섹션은 엘리먼트의 content 내의 문자 데이터 어디에나 삽입할 수 있음
    - 단, XML 마크업 내에서는 사용할 수 없음

XML 속성

- **속성(attributes)**
  - 엘리먼트에 대한 부가적인 정보나 데이터를 표현하기 위한 방법
  - 하나의 엘리먼트가 여러 개의 속성들을 가질 수 있음
  - 형식: 시작 태그의 일부로 표현

```
<element_name  attribute1="value1" attribute2="value2"... >
```

  - 구성요소: 속성명="속성값" 으로 표현

```
<student sid="100">
```

엘리먼트명 속성명 속성값

XML 속성

- 주의 사항
  - 속성은 반드시 속성 값을 가져야 함
    - 빈 문자열을 포함할 수도 있음 ("")
  - 속성값은 큰 따옴표(")나 작은 따옴표(')를 사용해야 함
  - 속성명 부여 방법은 엘리먼트의 태그명 부여 방법과 동일
    - 대소문자를 구별
    - 'xml'이라는 문자열로 시작할 수 없음
    - 숫자로 속성명을 시작할 수 없음
  - 주의: 하나의 엘리먼트에 같은 이름의 속성을 두 개 이상 선언할 수 없음

XML 속성

- 엘리먼트 vs. 속성

```
<student sid="100">
   <name>홍길동</name>
   <age>30</age>
   <address>서울시 성북구 월곡동</address>
   <phone type = "home">02-123-2345</phone>
   <phone type = "office">031-777-9999</phone>
</student>
```

```
<student sid="100">
   <name> 홍길동 </name>
   <age>30</age>
   <address> 서울시 성북구 월곡동 </address>
   <phone>
        <home>02-123-2345</home>
        <office>031-777-9999</office>
        <mobile>010-222-3333</mobile>
    </phone>
</student>
```

```
<student sid="100"
        name="홍길동
        age="30"
        address="서울시 성북구 월곡동 "
        phone="02-123-2345"
        phone="031-777-9999">
</student>
```

```
<student sid="100"
        name="홍길동
        age="30"
        address="서울시 성북구 월곡동 "
        h_phone="02-123-2345"
        o_phone="031-7777-9999"
        m_phone="010-222-3333">
</student>
```


주석

- 주석(comment)은 XML 문서를 작성하는 사람과 이용하는 사람이 좀더 쉽게 문서의 내용을 이해할 수 있도록 덧붙인 설명

```
<!-- 주석의 내용 -->
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- 루트 엘리먼트 -->
<booklist>
  <!--
  <book>
    <title>XML 기초서</title>
      <publisher>&pub1;</publisher>
  </book>
  -->
  <book>
    <title> <!-- 알기 쉬운 --> XML 기초서</title>
      <publisher>&pub1;</publisher>
  </book>
</booklist>
```


처리 명령어

- 처리 명령어(**Processing Instruction; PI**)
  - 해당 XML 문서를 처리하는 응용 프로그램(application)에게 XML 문서의 처리 방법을 지시함
  - 형식

```
<?name_processor instruction ?>
```

    - name_processor: 명령문이 전달되는 응용 프로그램 식별자
      - ✓ '<?'와 'name_processor' 사이에 공백문자를 포함할 수 없음
    - instruction: 응용 프로그램이 어떻게 문서를 처리할지를 나타냄
  - 예: CSS, XSL

```
<?xml-stylesheet  type="text/css" href="student_style.css" ?>
```


처리 명령어

- 주의 사항
  - name_processor 이름은 엘리먼트 이름과 동일한 규칙을 사용
    - 반드시 문자 또는 _로 시작해야 하고, 그 뒤에 숫자, 문자, . , _ 등을 자유롭게 사용할 수 있음
  - PI는 XML 문서의 어디든지 삽입할 수 있음
    - 단, 주석과 마찬가지로 마크업 내에는 삽입할 수 없음
    - 일반적으로 문서의 서두(prolog) 부분에서 사용됨

References

- Extensible Markup Language (XML) 1.0 (Fifth Edition)
  W3C Recommendation, 26 November 2008
  - http://www.w3.org/TR/REC-xml/
- XML Tutorial
  - http://www.w3schools.com/xml/
- Unicode & UTF-8
  - http://www.unicode.org/standard/WhatIsUnicode.html
  - http://en.wikipedia.org/wiki/Unicode
  - https://unicode-table.com/kr/
  - https://www.utf8-chartable.de/unicode-utf8-table.pl