

I. HTML Basics

1. Introduction to HTML

1) Why learn HTML?

모든 웹페이지는 HTML이라는 언어로 작성된다. HTML은 모든 웹페이지 구조의 골격이다. 여기서는 HTML을 사용하여 웹페이지에 paragraphs, headings, images, links를 추가하는 방법에 대하여 설명한다.

test.html라는 파일을 만들어보자. 이것은 우리의 첫 번째 HTML 파일이다. <> 속에 들어있는 코드를 보라. HTML 역시 다른 특별한 구분체계를 가지고 있다.

test.html

1. <!DOCTYPE html>
2. Feel free to change this text.

>Instructions

01. test.html file을 만든다.
02. 2번째 줄의 텍스트를 변경하라.(the bit between and)
03. 저장한 다음에 브라우저를 통해 tags 사이의 텍스트가 강조체로 바뀐 것을 확인하라.

2) HTML and CSS

HTML란 HyperText Markup Language의 준말이다. Hypertext란 "text with links in it." 라는 의미이다. 새로운 웹페이지로 가기 위해 어떤 단어를 클릭한다면, 여러분은 하이퍼텍스트를 클릭한 것이다.

markup language은 한 페이지에서 더 많은 일을 할 수 있도록 하기 위해 사용되는 프로그래밍 언어이다. 이것은 텍스트를 images, links, tables, lists 등으로 변경시킨다. HTML은 이러한 markup language 이다.

웹페이지를 예쁘게 만드는 것은 무엇인가? 그것은 바로 CSS-Cascading Style Sheets 이다. 이것은 HTML의 골격에 덧발라지는 피부 화장과 같다고 생각하라. 먼저 HTML을 배우고, 그 다음에 CSS를 배우기로 한다.

여러분이 첫 번째 해야할 일은 페이지의 골격을 갖추는 것이다.

01. 항상 첫 번째 줄에 `<!DOCTYPE html>`을 입력한다. 이것은 브라우저에게 읽을 언어가 무엇인지를 알려주어 주는 것이다(이 경우에는 HTML).
02. 그 다음 줄에는 항상 `<html>`을 입력한다. 여기서 HTML document는 시작된다.
03. 마지막 줄에는 항상 `</html>`을 입력한다. 여기서 HTML document는 끝난다.

>Instructions

01. 위의 test.html에서 , 두 번째와 마지막 줄에 the `<html>` 그리고 `</html>`을 입력하라.

3) Basic terminology

- Things inside `<>`s are called tags.
- Tags nearly always come in pairs: an opening tag and a closing tag.
Example of opening tag: `<html>`
Example of closing tag: `</html>`

여러분은 태그를 괄호와 같다고 생각하라. 여러분이 열 때마다 닫아야 한다. 태그는 또한 등지(nest)이므로 여러분은 올바른 순서로 그것을 닫아야 한다. 가장 최근에 연 태그가 첫 번째로 닫혀야 한다.

>example:

```
<first tag><second tag>Some text!</second tag></first tag>
```

이 모든 태그는 반드시 `<html>`와 `</html>` 태그 사이에 있어야 한다.

4) Make the head

HTML 파일의 모든 것은 opening `<html>`와 closing `</html>` tags 사이에 있어야 한다.

```
<!DOCTYPE html>  
<html>
```

```
</html>
```

HTML file은 항상 두 부분으로 구성된다: head 와 body. head를 시작해 보자. head는 타이틀과 같은 HTML 파일에 대한 정보가 포함된다. 여러분이 브라우저의 타이틀 바나 페이지 탭에서 보는 것이 바로 타이틀 이다.

>Instructions

```
<html>
  <head>
    <title>My Webpage</title>
  </head>
</html>
```

5) Paragraphs in the body

body에는 text, images, links와 같은 콘텐츠가 포함된다. 바디에 있는 콘텐츠는 실재 페이지에 나타나게 된다.

바디는 <html> tags 안에 있으며, <head> tags 바로 다음에 입력한다:

```
<html>
  <head>
    <title>My Webpage</title>
  </head>

  <body>

  </body>
</html>
```

>Instructions

01. 마감 </head> tag 아래에, 시작 <body> tag를 입력한 다음에 위의 예처럼 마감 </body> tag를 입력하라.
02. 바디 내부에, 두 개의 문장을 작성하라. 각 문장은 시작 <p> tag와 마감 </p> tag로 감싸야 한다. 아래의 예처럼 콘텐츠를 작성하라:

```
<body>
  <p>Hello world!</p>
  <p>This is my second paragraph</p>
</body>
```

>Hint

이제 여러분의 HTML 문서는 다음과 같을 것이다:

```
<html>
  <head>
    <title>My Webpage</title>
  </head>

  <body>
    <p>This is my first paragraph</p>
    <p>This is my second paragraph</p>
  </body>
</html>
```

2. Body elements

6) Paragraphs and headings

heading tags용으로 우리의 . paragraphs headings을 사용하기 위하여 <h1> tag를 사용해 보자. 이 태그 사이의 콘텐츠가 가장 크게 나타날 것이다.

heading size는 6가지가 있으며, <h1>이 가장 크고, <h6>가 가장 작다.

```
<h1> - The CEO
<h2> - VP
<h3> - Director
<h4> - Middle management
<h5> - Lowly assistant
<h6> - Gets coffee for everyone
```

3. Adding images to your site!

7) You're going places!

다른 웹 사이트나 페이지로 가려면, 다음과 같이 하이퍼링크를 사용하여야 한다.

```
<a href="http://www.daegu.ac.kr">My University Site!</a>
```

01. 시작 <a> tag를 입력한 다음에 그 태그가 href라 부르는 속성을 갖도록 한다. href의 값에는 예를 들어 http://www.daegu.ac.kr과 같이 자신이 가고자 하는 곳의 링크를 표기한다.

02. 그런 다음에, 시작 <a>와 마감 tags 사이에 해당 링크의 설명 텍스트를 입력한다. 이 텍스트가 링크용으로 클릭된다.

03. 최종적으로 마감 tag를 입력한다.

11) Adding images

image tag는 이다. 이 태그는 다른 태그와는 좀 다르다. 태그 사이에 콘텐츠를 입력하는 대신에, 이 태그는 src 를 사용하여 그림을 얻는 장소를 알려준다. 이것은 또한 마감 태그가 없다는 것에 주의하여야 한다. 이것은 마감하기 위하여 그 태그 안에 /를 입력한다:

```

```

>Instructions

원숭이 그림보기:

http://image.babosarang.co.kr/product/detail/DYG/1209191617014339/_500.jpg

12) Click that image

이미지로 링크 만들기의 예:

```
<a href="http://www.daegu.ac.kr/">
```

```
<img src=
```

```
"http://image.babosarang.co.kr/product/detail/DYG/1209191617014339/\_500.jpg"
```

```
/>
```

```
</a>
```

>Hint

완전한 웹 어드레스를 입력하라. 그리고 인용부로 그 주소를 감싸라. 여러분의 코드는 다음과 같이 보여야 한다:

```
<a href="LINK url">
  
</a>
```

***Build your Own Webpage!!**

예: Index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Result</title>
  </head>
  <body><h1>YEAH SANDWICHES</h1>
  
    <p>I like eggs.</p>
    <p>And ham!</p>
    <p>But mostly sandwiches.</p>
  </body>
</html>
```

4. Introduction to CSS

The stylesheet.css tab에는 모든 CSS styling information가 포함되어 있다: HTML elements가 가야할 곳, 표현해야할 색깔, 그것들의 크기 등등.

CSS (Cascading Style Sheets)는 HTML의 모습을 나타내고 포맷팅하는데 사용되는 언어이다. style sheet는 HTML 파일의 모습을 기술하는 파일이다.

>예:

```

<!DOCTYPE html>
<html>
  <head>
    <title>Arduino SD Card Web Page</title>
    <style type="text/css">
      h1 {
        font-family: courier, courier-new, serif;
        font-size: 20pt;
        color: blue;
        border-bottom: 2px solid blue;
      }
      p {
        font-family: arial, verdana, sans-serif;
        font-size: 12pt;
        color: #6B6BD7;
      }
      .red_txt {
        color: red;
      }
    </style>
  </head>
  <body>
    <h1>Arduino SD Card Page with CSS</h1>
    <p>Welcome to the Arduino web page with CSS styling.</p>
    <p class="red_txt">This text is red.</p>
    <p>This paragraph has one word that uses <span class="red_txt">red</span> text.</p>
  </body>
</html>

```

>>결과:



1) Why separate form from function?

functional content/structure (HTML)와 form/formatting (CSS)를 분리시키는 두 가지의 중요한 이유:

01. 재작성 없이 여러개의 HTML elements(e.g. style="color:red";)에 동일한 포맷을 적용시킨다.
02. 단일 CSS file을 사용한 여러 개의 HTML 페이지들이 유사한 모습과 포맷팅을 갖도록 한다.

2) Link it up!

<link> tag는 3 가지의 속성이 필요하다:

01. A **type** attribute that should always be equal to "text/css"
02. A **rel** attribute that should always be equal to "stylesheet"
03. A **href** attribute that should point to the web address of your CSS file

예):

```
<link type="text/css" rel="stylesheet" href="stylesheet.css"/>
```

5. Table 만들기

```
<!DOCTYPE html>
<html>
<head>

<style>
table, tr, td {
    border: 1px solid black;
}
</style>

</head>

<body>

<h2>HTML Table Sample</h2>

<table style="width:100%">
  <tr>
    <th>Title</th>
    <th>Creator</th>
    <th>Pub_Year</th>
  </tr>
  <tr>
    <td>Library and Big Data</td>
    <td>Maria Anders</td>
    <td>2019</td>
  </tr>
```



```

<tr>
  <td>Library Technology</td>
  <td>Francisco Chang</td>
  <td>2017</td>
</tr>
<tr>
  <td>Digital Library</td>
  <td>Roland Mendel</td>
  <td>2016</td>
</tr>
</table>

</body>

</html>

```

II. JavaScript 맛보기

Sample 1:

```

<!DOCTYPE html>
<html>
<body>
<h1>My First JavaScript</h1>
<p>Click Date to display current day, date, and time.</p>
<button type="button" onclick="myFunction()">Date</button>
<p id="demo"></p>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML = Date();
}
</script>
</body>
</html>

```

Sample 2:

```

<!DOCTYPE html>
<html>
<body>
<p id="p1">Please locate where 'locate' occurs!</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var str = document.getElementById("p1").innerHTML;
    var pos = str.indexOf("locate");
    document.getElementById("demo").innerHTML = pos;
}
</script>
</body>
</html>

```

1) Writing Into HTML Output

```

<!DOCTYPE html>
<html>
<body>
<p>
JavaScript can write directly into the HTML output stream:
</p>
<script>
document.write("<h1>This is a heading</h1>");
document.write("<p>This is a paragraph.</p>");
</script>
<p>

```

You can only use `document.write` in the HTML output. If you use it after the document has loaded (e.g. in a function), the whole document will be overwritten.

```

</p>
</body>
</html>

```

2) Reacting to Events

```
<button type="button" onclick="alert('Welcome!')">Click Me!</button>
```

- * The alert() function is not much used in JavaScript, but it is quite handy for trying out code.
- * The onclick event is only one of the many HTML events you will learn about in this tutorial.

3) Changing HTML Content

```
<p id="demo">
```

JavaScript can change the content of an HTML element.

```
</p>
```

```
<script>
```

```
function myFunction()
```

```
{
```

```
x=document.getElementById("demo"); // Find the element
```

```
x.innerHTML="Hello JavaScript!";    // Change the content
```

```
}
```

```
</script>
```

```
<button type="button" onclick="myFunction()">Click Me!</button>
```

- * You will often see document.getElementById("some id"). This is defined in the HTML DOM. The DOM (Document Object Model) is the official W3C standard for accessing HTML elements.

4) Changing HTML Images

```
<script>
```

```
function changeImage()
```

```
{
```

```
element=document.getElementById('myimage')
```

```
if (element.src.match("bulbon"))
```

```
{
```

```
    element.src="pic_bulboff.gif";
```

```
}
```

```

else
{
    element.src="pic_bulbon.gif";
}
}
</script>



<p>Click the light bulb to turn on/off the light</p>

```

III. XML Basic

1) XML이란?

- XML stands for EXtensible Markup Language
- XML is a markup language much like HTML
- XML was designed to store and transport data
- XML was designed to be self-descriptive
- XML is a W3C Recommendation

2) XML Does Not DO Anything

아마도 이것을 이해하기는 좀 힘들겠지만, XML은 결코 어떤 것도 하지 않는다.

다음의 note는 XML로 저장된 Jani가 Tove에 보낸 노트 이다:

예:

```

<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>

```

이 노트는 매우 자기-기술적(self-descriptive)이다. 이것은 보내는 사람과 받는 사람에 대한 정보를 갖고 있다. 또한 하나의 헤딩과 하나의 메시지 바디도 갖고 있다.

그러나 아직까지 이 XML document는 어떠한 일도 하지 않는다. XML은 단지 태그에 감싸져 있는 정보 이다. 누군가 그것을 보내거나, 받거나, 저장하거나, 보기 위해서는 한 조각의 소프트웨어를 작성해야만 한다.

>Note

To: Tove

From: Jani

Reminder

Don't forget me this weekend!

3) The Difference Between XML and HTML

XML과 HTML은 서로 다른 목적으로 디자인 되었다:

- XML은 데이터가 무엇인지에 초점을 맞추어 데이터 전달용으로 디자인 되었다.
- HTML은 데이터가 어떻게 보여야 하는지에 초점을 맞추어 데이터 디스플레이용으로 디자인 되었다.
- XML tags는 HTML tags처럼 미리 규정되어 있지 않다.

4) XML Does Not Use Predefined Tags

XML 언어는 어떠한 미리 규정된 태그도 갖고 있지 않다.

위의 예(like <to> and <from>)에 있는 태그들은 are not defined in any XML 기준에 정의되어 있지 않다. 이들 태그들은 the XML document의 저자에 의해 만들어진 것이다.

HTML은 <p>, <h1>, <table>, etc처럼 미리 정해진 태그를 사용한다.

XML에서, 저자는 tags 와 document structure 둘 다를 정의해야 한다.

5) XML is Extensible

대부분의 XML applications은 새 데이터가 추가되거나 제거될 때조차도 작업할 수 있다.

note.xml (<to> <from> <heading> <data>)의 오리지널 버전을 디스플레이하도록 디자인된 어플에 대해 상상해 보자.

그런 다음에 <date>와 <hour> elements와 제거된 <heading>을 추가한 새로운 버전의 note.xml에 대해 상상해 보자.

XML을 구축할 때, 이전 버전이 아직 작동하고 있을 것이다:

```
<note>
  <date>2015-09-01</date>
  <hour>08:30</hour>
  <to>Tove</to>
  <from>Jani</from>
  <body>Don't forget me this weekend!</body>
</note>
```

>>결과:

Note

To: Tove

From: Jani

Date: 2015-09-01 08:30

Head: (none)

Don't forget me this weekend!

6) XML Simplifies Things

- data sharing을 간단하게 한다.
- data transport을 간단하게 한다.
- platform changes를 간단하게 한다
- data availability를 간단하게 한다.

많은 컴퓨터 시스템이 호환할 수 없는 포맷으로 데이터를 갖고 있다. 비-호환 시스템 간에 데이터를 교환하거나 업-그레이딩하는 것은 매우 시간 소모적인 일이다. 대량의 데이터가 변환되어야 하며, 종종 호환불가 데이터는 분실되기도 한다.

XML은 plain text format으로 데이터를 저장한다. 이것은 데이터를 저장, 운송, 공유하는데 있어서 소프트웨어- 그리고 하드웨어- 독립적인 방법을 제공한다.

XML은 또한 새 운영 시스템, 새 어플, 또는 새 브라우저에서 데이터 손실없이 확장과 업

그래이드를 보다 쉽게 하도록 한다.

XML로 된 데이터는 people, computers, voice machines, news feeds 등과 같은 모든 종류의 "reading machines"에서 이용할 수 있다.

7) XML Separates Data from Presentation

XML은 디스플레이 방법에 대한 어떠한 정보도 전달하지 않는다.

동일한 XML data는 서로 다른 많은 presentation scenarios에서 사용할 수 있다.

이런 이유로, XML은 data와 presentation 사이에서 완전히 독립적이다.

8) XML Separates Data from HTML

HTML에서 데이터를 디스플레이할 때, 그리고 그 데이터가 변경될 때, HTML file을 편집하지 않아야 한다. when the data changes.

XML에서 데이터는 독립된 XML files에 저장된다.

두세 줄의 JavaScript code를 갖고 있는, XML file을 읽고 HTML page의 데이터 콘텐츠를 경신할 수 있다.

9) XML Tree Structure

XML documents는 element trees처럼 형성되어 있다.

XML tree는 starts at a root element에서 시작되고, branches는 the root에서부터 child elements까지 발생한다.

모든 elements는 sub elements (child elements)를 가질 수 있다:

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

parent, child, and sibling 용어는 엘리먼트 간의 관계를 설명하는데 사용된다.

Parent는 children을 갖는다. Children은 parents를 갖는다. Siblings은 동일한 수준의 children(brothers and sisters) 이다.

모든 elements는 text content (Harry Potter) and attributes (category="cooking")을 가질 수 있다.

10) Self-Describing Syntax

XML은 많은 self-describing syntax를 사용한다.

prolog에서는 XML version 그리고 character encoding을 정의한다:

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

그 다음의 줄은 도큐먼트의 root element 이다:

```
<bookstore>
```

그 다음의 줄에서 <book> element가 시작된다:

```
<book category="cooking">
```

<book> elements는 4 child elements를 갖는다: <title>, < author>, <year>, <price>.

```
<title lang="en">Everyday Italian</title>
<author>Giada De Laurentiis</author>
<year>2005</year>
<price>30.00</price>
```

그 다음 줄에서 book element를 마감한다:

```
</book>
```

위의 예를 통하여, 우리는 XML document가 서점에 있는 책에 대한 정보를 갖고 있다고 추측할 수 있다.

11) An Example XML Document

Book.xml

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

```
<?xml-stylesheet type="text/xsl" href="book.xsl"?>
```


<bookstore>

```
<book category="cooking">
  <title lang="en">Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>
```

```
<book category="children">
  <title lang="en">Harry Potter</title>
  <author>J. K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>
```

```
<book category="web">
  <title lang="en">XQuery Kick Start</title>
  <author>James McGovern</author>
  <author>Per Bothner</author>
  <author>Kurt Cagle</author>
  <author>James Linn</author>
  <author>Vaidyanathan Nagarajan</author>
  <year>2003</year>
  <price>49.99</price>
</book>
```

```
<book category="web" cover="paperback">
  <title lang="en">Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
</book>
```

</bookstore>

book.xsl

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
```

```

<xsl:template match="/">
  <html>
    <body>
      <h2>My First Book Collection:</h2>
      <table border="1">
        <tr bgcolor="#9acd32">
          <th>Title</th>
          <th>Author</th>
          <th>Year</th>
          <th>price</th>
        <xsl:for-each select="bookstore/book">
          <tr>
            <td>
              <xsl:value-of select="title" />
            </td>
            <td>
              <xsl:value-of select="author" />
            </td>
            <td>
              <xsl:value-of select="year" />
            </td>
            <td>
              <xsl:value-of select="price" />
            </td>
          </tr>
        </xsl:for-each>
      </table>
    </body>
  </html>
</xsl:template>
</xsl:stylesheet>

```

>>결과: <XML Notepad 2007> 열어서 보기

IV. RDF 맛보기

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="rdf.xsl"?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc="http://purl.org/dc/elements/1.1/">

<rdf:Description rdf:about="http://en.wikipedia.org/wiki/Oxford">

<dc:title>Oxford</dc:title>
<dc:coverage>Oxfordshire</dc:coverage>
<dc:publisher>Wikipedia</dc:publisher>

</rdf:Description>

</rdf:RDF>

```

RDF 보기:

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body style="font-family:Arial;font-size:12pt;background-color:#EEEEEE">
        <xsl:for-each select="rdf:RDF/rdf:Description">
          <div style="background-color:gold;color:blue;padding:20px">
            <xsl:value-of select="dc:title"/>
            <p/>
            <xsl:value-of select="dc:coverage"/>
            <p/>
            <xsl:value-of select="dc:publisher"/>
          </div>
        </xsl:for-each>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>

```

FIN