CHAPTER

02 웹 프로그래밍 기초

이 장에서 다룰 내용

- URL
- 웹 브라우저와 웹 서버
- HTML과 HTTP
- JSP 프로그래밍

웹 프로그래밍을 하려면 웹에 대한 이해가 필요하다. 이 장에서는 JSP를 본격적으로 학습하기에 앞서 웹이 무엇이고, 웹과 관련된 주요 구성 요소에 대해 살펴보도록 하자. 또한, 간단한 코드를 통해 첫 번째 JSP 웹 프로그래밍을 만들어볼 것이다.

01 웹과 웹 프로그래밍

웹 브라우저에 http://javacan.tistory.com/390을 입력하면 [그림 2.1]과 같은 결과가 화면에 표시되는 것을 알 수 있다.

[그림 2.1] 웹 브라우저를 이용해서 웹을 사용

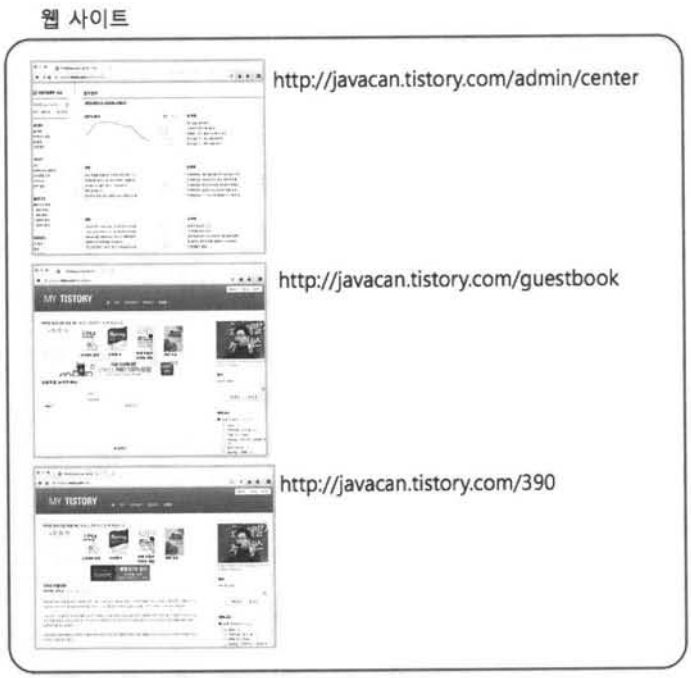
일반 사용자는 웹 브라우저에 단순히 http://javacan.tistory.com/390이나 http://www.daum.net과 같은 주소를 입력하면 웹을 사용할 수 있지만, 웹을 만들어야 하는 개발자는 알아야 할 것들이 더 많다. 예를 들어, http://www.daum.net과 같은 주소를 뭐라고 부르는지, http://www.daum.net이란 주소를 입력하면 웹 브라우저에 어떻게 내용들이 출력되는지에 대한 기초적인 내용 정도는 알고 있어야 한다. 이번 절에서는 이렇게 개발자가 기본적으로 알아야 할 내용을 설명한다.

1.1 URL과 웹 페이지

http://javacan.tistory.com/390이나 http://www.11st.co.kr/html/main.html과 같이 웹 브라우저의 주소줄에 표시되는 것을 URL이라고 부른다. URL은 Uniform Resource Locator의 약자로 이는 일종의 주소와 같은 역할을 한다. 집 주소가 다르면 다른 건물인 것처럼 URL이 다른 경우에도 [그림 2.2]와 같이 다른 결과가 웹 브라우저에 표시되는 것을 알 수 있다.

[그림 2.2] URL은 웹 페이지의 주소

웹 브라우저의 주소줄에 URL을 입력하면 웹 브라우저에 URL에 해당하는 내용이 출력되는데, 이렇게 웹 브라우저에 출력된 내용을 웹 페이지(web page)라고 부른다. 흔히 홈페이지라고 부르는 웹 사이트는 이런 웹 페이지의 묶음이다.



[그림 2.3] 웹 페이지의 묶음이 웹 사이트가 된다.

웹 페이지의 주소를 표현할 때 사용하는 URL은 몇 개의 요소로 구성되는데, 주요 구성 요소는 [그림 2.4]와 같다.

[그림 2.4] URL의 일반적인 구성

URL의 주요 구성 요소에 대한 내용을 [표 2.1]에 정리했다.

표 2.1 URL의 주요 구성 요소

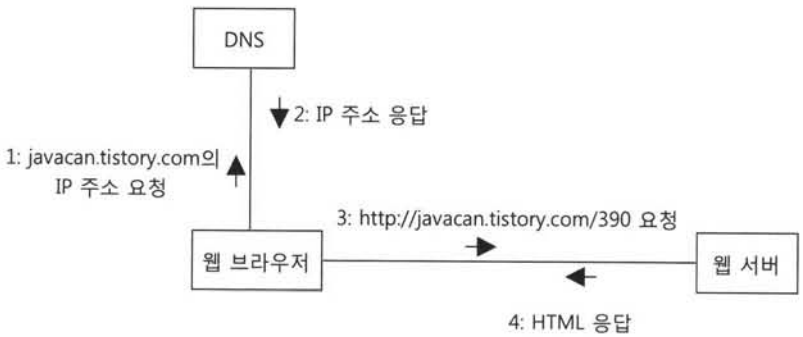
구성 요소	설명
프로토콜	웹 브라우저가 서버와 내용을 주고받을 때 사용할 규칙 이름이다. 웹 페이지의 주소를 표현할 때는 http를 사용한다.
서버 이름	웹 페이지를 요청할 서버의 이름을 지정한다. 서버 이름은 "javacan.tistory.com"과 같은 도메인 이름이나 180.70.134.239와 같은 IP 주소를 입력할 수 있다.
경로	웹 페이지의 상세 주소에 해당한다. 즉, 웹 페이지마다 다른 경로를 갖는다.
쿼리 문자열	추가로 서버에 보내는 데이터에 해당한다. 같은 경로라 하더라도 입력한 값에 따라 다른 결과를 보여줘야 할 때 쿼리 문자열을 사용한다. 예를 들어, 검색 결과를 보여주는 페이지를 생각해보자. 이 페이지는 입력한 검색어에 따라 다른 내용이 표시되는데, 보통 쿼리 문자열을 이용해서 검색어를 전달한다.

NOTE

프로토콜 이름과 서버 이름 사이에 위치한 "://"은 프로토콜과 나머지 부분을 구분하기 위해 사용된다. URL을 좀더 범용적으로 정의한 URI도 있는데, URL과 URI에 대한 내용은 각각 http://ko.wikipedia.org/wiki/URL과 http://goo.gl/1gNfgx 페이지를 참고하기 바란다.

1.2 웹 브라우저와 웹 서버

http://javacan.tistory.com/390과 같은 URL을 입력했을 뿐인데 어떻게 [그림 2.1]과 같은 화면이 웹 브라우저에 표시되는 걸까? 웹 브라우저에 URL에 해당하는 웹 페이지가 출력되기까지 꽤 복잡한 과정을 거치는데, 이 과정을 단순화하면 [그림 2.5]와 같이 정리할 수 있다.



[그림 2.5] 웹 브라우저와 웹 서버의 통신 과정

웹 브라우저에 URL을 입력하면 웹 서버라 불리는 프로그램이 웹 브라우저에 웹 페이지를 제공한다. [그림 2.5]에서 4번 과정이 웹 서버가 웹 브라우저에 웹 페이지를 제공하는 단계이다. 웹 브라우저가 웹 서버에 웹 페이지를 달라고 하는 것을 흔히 '요청(request)'한다고 표현하고, 요청한 웹 페이지를 웹 브라우저에 제공하는 것을 '응답(response)'이라고 표현한다.

웹 브라우저와 웹 서버는 다른 컴퓨터에 위치한다. 예를 들어, 집의 PC에서 웹 브라우저에 http://www.daum.net 주소를 입력할 때 연결하는 웹 서버 프로그램은 집 PC가 아닌 다른 컴퓨터에서 실행되고 있다. 웹 서버가 다른 컴퓨터에서 실행되고 있기 때문에, 웹 브라우저가 웹 서버에 연결하려면, 웹 서버가 실행중인 컴퓨터의 주소를 알아야 하는데, 이 주소를 IP 주소라고 부른다.

휴대폰의 전화번호가 휴대폰마다 다른 것처럼, 인터넷을 통해 연결되는 컴퓨터들도 고유의 주소 값인 IP 주소를 갖고 있다. 그런데, IP 주소는 180.70.134.239와 같은 숫자로 구성되어 있기 때문에 외우기가 쉽지 않다. 이런 이유로 IP 주소 대신에 "javacan.tistory.com"과 같이 사람이 기억하기 좋은 도메인 이름을 사용한다.

웹 브라우저와 웹 서버는 IP 주소를 이용해서 연결하기 때문에 도메인 이름을 IP 주소로 변환할 필요가 있는데, 이 때 사용하는 것이 바로 DNS(Domain Name Server)이다. 웹 브라우저에서 URL을 입력하면, 웹 브라우저는 [그림 2.5]의 1번 과정처럼 도메인 이름에 해당하는 IP 주소를 DNS에 요청하고, DNS는 2번 과정처럼 IP 주소를 응답으로 제공한다.

DNS로부터 IP 주소를 받으면, 웹 브라우저는 3번 과정처럼 IP 주소를 이용해서 웹 서버에 연결한 뒤 URL에 해당하는 웹 페이지를 요청하고, 웹 페이지를 응답으로 받게 된다.

> **용어 클라이언트와 서버**
>
> 일반적으로 네트워크 프로그램에서 요청하는 쪽을 클라이언트(Client)라고 부르고, 요청을 받아 알맞은 기능이나 데이터를 제공하는 쪽을 서버(Server)라고 부른다. 웹 브라우저는 HTML 문서나 이미지 등을 요청하므로 클라이언트에 해당하고, 웹 서버는 이름에서 유추할 수 있듯이 서버에 해당된다.
>
> 서버 프로그램이 실행되는 컴퓨터를 서버라고도 부른다. 일반적으로 서버 컴퓨터는 가정에서 사용하는 컴퓨터에 비해 뛰어난 성능과 안정성을 제공한다.

한 개의 컴퓨터에는 웹 서버 프로그램만 실행되는 것이 아니다. 웹 서버 프로그램과 함께 동영상을 제공해주는 스트리밍 서버 프로그램도 실행될 수 있고, 채팅 서비스를 위한 채팅 서버 프로그램이 실행될 수도 있다. IP 주소는 연결할 컴퓨터를 구분하는데 사용되기 때문에, IP 주소만으로는 컴퓨터의 어떤 서버 프로그램을 실행할지 알 수 없다.

이런 이유로 각 서버 프로그램은 클라이언트가 연결할 때 다른 서버 프로그램과 구분할 수 있도록 포트(port)라는 것을 사용한다. 포트는 숫자로 된 번호로서 서버 프로그램마다 구분되는 포트 번호를 사용하며, 클라이언트는 IP 주소와 함께 포트 번호를 사용해서 원하는 서버 프로그램에 연결하게 된다. 앞서 1장에서 톰캣이 정상적으로 실행되는지 확인하기 위해 "http://localhost:8080/index.jsp"를 URL로 입력했는데, 이때 콜론(":") 다음에 위치한 8080이 포트 번호가 된다.

웹 서버가 사용하는 기본 포트 번호는 80이다. 이는 URL에서 프로토콜이 http인 경우 별도로 포트를 지정하지 않으면 80 포트로 연결한다는 것을 뜻한다. 즉, "http://localhost/"라는 URL을 사용하면 웹 브라우저는 80 포트를 이용해서 서버에 연결한다. 톰캣은 기본적으로 8080 포트를 사용하도록 설정되어 있기 때문에, URL에 포트 번호를 붙여 톰캣 서버를 테스트한다.

NOTE

톰캣의 설정을 변경하면 8080 포트 대신에 웹 서버의 기본 포트인 80 포트를 사용할 수 있다. 톰캣을 설치한 폴더의 conf/server.xml 파일을 보면 8080 포트 번호를 지정하는 설정 부분을 찾을 수 있을 것이다. 이 설정 값을 80으로 바꿔주면 톰캣을 80 포트로 실행할 수 있다.

1.3 HTML과 HTTP

웹 브라우저에서 http://www.daum.net에 연결한 뒤에 내용 영역에서 마우스 오른쪽 버튼을 눌러 표시되는 단축 메뉴에서 [소스 보기] 메뉴를 선택하여 실행해보자. 그러면, 다음과 비슷하게 문자로 구성된 긴 코드를 볼 수 있을 것이다. (매우 길기 때문에 많은 부분을 생략했다.)

```
<!DOCTYPE html>
<html lang="ko">
  <head>
    <meta charset="utf-8">
    <title>Daum &ndash; 모으다 잇다 흔들다</title>
    ...생략
  <body>
    <div id="daumIndex" class="d_index">
      <a id="mainServiceDirectLink" ...생략>
        <span class="img_vert inner_shortcut">주요 서비스 바로가기</span>
      </a>
    ...생략
    ...생략
</body>
</html>
```

위 코드는 HTML(HyperText Markup Language)이라고 불리는 표준을 이용해서 작성한 것이다. 웹 페이지를 만들 때 사용하는 것이 바로 HTML이며, HTML 표준에 정의된 <html>, <body>, <a> 등의 구성 요소를 이용해서 웹 페이지를 작성하게 된다. HTML을 이용해서 작성했다고 해서 HTML 문서라고도 부른다.

웹 서버는 URL에 해당하는 HTML 문서를 전송하는데, HTML 문서를 받은 웹 브라우저는 정해진 규칙에 따라 HTML 문서를 분석해서 알맞은 화면을 생성한다. HTML 표준에 따라 HTML 문서로부터 알맞은 화면을 생성하는 과정을 렌더링(rendering)이라고도 표현한다.

[그림 2.6] 웹 브라우저는 HTML 문서를 분석해서 알맞은 화면을 보여준다.

NOTE

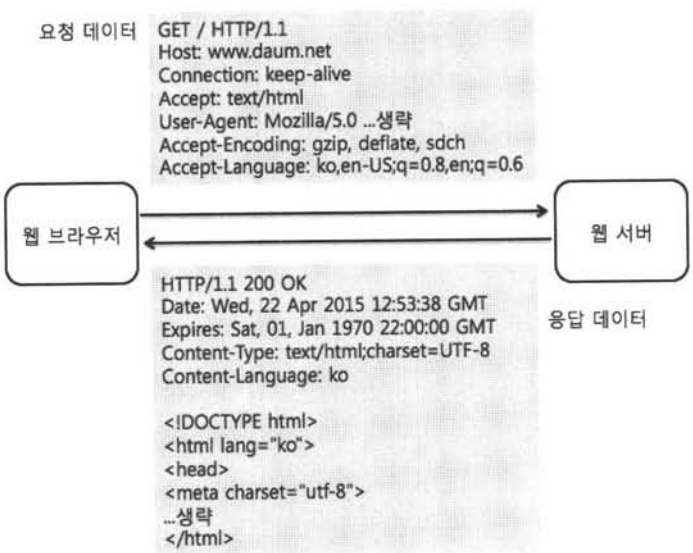
이 책에서는 비교적 간단한 HTML만 사용한다. 기본적인 HTML 문법은 1주일 정도 학습하면 익힐 수 있으니, HTML을 잘 모르는 독자는 꼭 HTML 기초를 익히도록 하자.

소포를 주고받을 때 전달하려는 내용물을 상자에 담고 상자 겉에 내용물에 대한 내용을 적어 보내는 것처럼 HTML 문서(즉, 웹 페이지)도 HTTP라는 방식의 상자를 이용해서 전송한다. HTTP는 Hypermedia Transfer Protocol의 약자로 웹 브라우저와 웹 서버가 HTML을 비롯해 이미지, 동영상, XML 문서 등 다양한 데이터를 주고받을 때 사용하는 일종의 규칙이다.

HTTP는 크게 다음과 같이 두 가지 관점에서 규칙을 정의하고 있다.

- **요청 규칙** :
  웹 브라우저가 웹 서버에 HTML과 같은 것을 요청할 때 사용할 데이터 구성 규칙
- **응답 규칙** :
  웹 서버가 웹 브라우저에 HTML과 같은 것을 전송할 때 사용할 데이터 구성 규칙

예를 들어, http://www.daum.net이란 주소를 웹 브라우저에 입력하면 실제로 웹 브라우저와 웹 서버가 주고받는 데이터는 [그림 2.7]과 같은 모양을 갖는다.



[그림 2.7] 웹 브라우저와 웹 서버는 HTTP를 이용해서 데이터를 주고받는다.

요청 데이터는 웹 브라우저가 웹 서버로부터 무엇을 받고 싶은지 기술한다. 이 요청 데이터를 받은 웹 서버는 요청 데이터에 기술한 정보를 이용해서 웹 브라우저가 요청한 것을 응답 데이터에 담아 보낸다.

HTTP에서 요청 데이터와 응답 데이터는 크게 '요청/응답 줄', '헤더', '몸체' 세 개의 영역으로 구성된다. 요청 데이터와 응답 데이터에서 첫 줄이 각각 '요청 줄'과 '응답 줄'이다. 예를 들어, [그림 2.7]에서 "GET / HTTP/1.1"이 요청 줄이고, "HTTP/1.1 200 OK"가 응답 줄에 해당한다. 헤더 영역은 "요청/응답 줄" 다음에 위치하며 "헤더이름: 헤더 값"으로 구성된 헤더 목록으로 구성된다. "Host: www.daum.net"의 경우 헤더 이름이 "Host"가 되고 헤더 값은 "www.daum.net"이 된다. 헤더가 끝난 다음에 빈 줄이 오고 그 다음에 몸체 내용이 온다.

HTTP 프로토콜의 요청 데이터와 응답 데이터에서 첫 줄, 헤더, 몸체가 갖는 데이터는 [표 2.2]와 같다.

표 2.2 HTTP의 요청/응답 데이터의 구성 요소

구성 요소	요청 데이터	응답 데이터
요청/응답 줄	GET이나 POST와 같은 HTTP 요청 방식(method)과 요청하는 자원의 경로를 지정한다.	요청에 대해 200이나 404같은 응답 코드를 전송한다. 참고로 200은 요청을 정상적으로 처리했음을 의미한다.
헤더	서버가 응답을 생성하는데 참조할 수 있는 정보를 전송한다. 예를 들어, 브라우저의 종류나 언어 등의 정보를 보낸다.	응답에 대한 정보를 전송한다. 응답의 몸체가 어떤 데이터인지, 길이는 어떻게 되는지 등에 대한 정보를 담는다.
몸체	정보를 전송해야 할 때 사용한다. 예를 들어, 파일 업로드와 같은 기능을 사용하면 몸체 영역에 파일을 담아 웹 서버에 전송한다.	웹 브라우저가 요청한 자원의 내용을 담는다. HTML 문서나 이미지 파일 데이터 등이 몸체 영역을 이용해서 전달된다.

[표 2.2]에서 '자원'이라는 단어를 사용했는데, 그 이유는 HTTP가 HTML 문서만 전송하는 용도로 사용되는 것이 아니기 때문이다. HTTP는 HTML 뿐만 아니라 이미지, 자바스크립트 코드 등 다양한 데이터를 전송하는데 사용되기 때문에, 이들을 포괄적으로 지칭하기 위해 '자원(resource)'이라는 단어를 사용하고는 한다.

NOTE

HTTP에 정의된 규칙은 꽤 복잡한데, 다행히 HTTP에 대해서 잘 알지 못해도 이 책을 학습하는데 문제는 없다. 하지만, 뛰어난 웹 프로그래머가 되려면 HTTP 프로토콜에 대해 어느 정도 이해하고 있어야 한다. HTTP에 대해 깊게 공부하고 싶다면, "HTTP 완벽 가이드" 책을 읽어볼 것을 권한다.

1.4 정적 자원과 동적 자원

웹 서버는 기본적으로 웹 브라우저가 요청한 경로를 분석한 뒤, 경로에 해당하는 파일을 읽어와 응답 데이터로 전송한다. 예를 들어, 1장에서 설치한 톰캣을 실행한 뒤 웹 브라우저에 http://localhost:8080/docs/index.html을 입력해보자. 이 경우, 톰캣은 [톰캣 폴더]/webapps/docs/index.html 파일의 내용을 응답 데이터로 웹 브라우저에 전송한다. 같은 URL을 웹 브라우저에 한 번 더 입력해도 동일한 결과가 화면에 표시된다.

톰캣을 포함해 많은 웹 서버들이 URL의 경로와 일치하는 파일을 읽어와 응답으로 전송하기 때문에, 파일이 바뀌기 전까지 웹 서버는 항상 같은 내용을 웹 브라우저에 전송한다. 따라서, 웹 브라우저는 (파일이 바뀌지 않았다면) 늘 같은 응답 데이터를 받으므로 동일한 화면을 출력한다. 이렇게 고정된 결과가 출력된다고 해서 이들 URL에 해당하는 자원을 정적(static) 페이지 또는 정적 자원이라고 표현한다. 보통 이미지 파일이나 HTML 파일과 같이 자주 바뀌지 않는 것들을 정적 자원으로 제공한다.

정적 자원과 달리 파일(코드)을 바꾸지 않아도 조건에 따라 다른 응답 데이터를 전송하는 경우도 있다. 예를 들어, 페이스북의 메시지 내용을 보여주는 주소인 https://www.facebook.com/messages URL을 입력하면 URL을 입력한 시점에 따라 다른 내용이 웹 브라우저에 표시된다. 비슷하게 구글은 쿼리 문자열에 따라 다른 검색 결과를 보여준다. 이렇게 시간이나 특정 조건에 따라 응답 데이터가 달라지는 자원을 동적(dynamic) 페이지 또는 동적 자원이라고 부른다. 이 책에서 배울 JSP를 비롯해서 PHP, ASP.net 등 많은 웹 관련 기술들이 바로 동적 페이지를 만드는데 사용되는 프로그래밍 기술이다.

1.5 웹 프로그래밍과 JSP

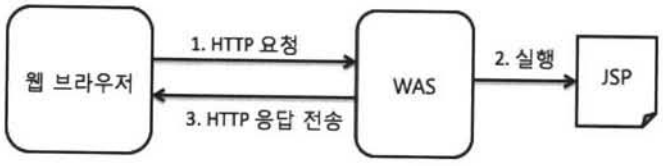
URL, 웹 서버, HTML, HTTP 등 웹 프로그래밍을 하는데 필요한 기본 지식에 대해 살펴봤다. 그렇다면, 웹 프로그래밍이란 무엇일까? 웹 프로그래밍이란 간단히 말하면 웹 서버가 웹 브라우저에 응답으로 전송할 데이터를 생성해주는 프로그램을 작성하는 것이다. 웹 프로그래밍을 하기 위해서 네트워크 처리, HTTP 헤더, 파일 입출력 처리 등에 대해서 알아야 할 필요는 없다. 이런 것들은 웹 서버에서 처리하며, 우리는 웹 서버가 실행하는 프로그램만 만들어주면 된다.

웹 서버의 종류에 따라 웹 프로그래밍을 할 때 사용할 기술이 달라진다. 예를 들어, 아파치 웹 서버를 사용하면 PHP라는 기술을 이용해서 웹 프로그래밍을 할 수 있고, 윈도우의 IIS 웹 서버를 사용하면 ASP.net이라는 기술을 이용해서 웹 프로그래밍을 할 수 있다. 이 책의 주제인 JSP 역시 웹 프로그래밍을 할 때 사용되는 기술이다.

JavaServer Pages, 줄여서 JSP는 동적 페이지를 작성하는데 사용되는 자바의 표준 기술로서 HTML 응답을 생성하는데 필요한 기능을 제공하고 있다. 더 정확하게는 HTML 응답뿐만 아니라 XML, JSON, 바이너리 파일 등도 응답으로 생성할 수 있지만, 주로 HTML 응답을 생성하는 목적으로 JSP를 사용한다. JSP를 잘 하려면 자바에 대한 이해가 필요하지만, JSP 자체의 문법이나 기능은 어렵지 않기 때문에 웹 개발을 이제 막 시작하는 개발자도 비교적 쉽게 JSP를 익힐 수 있다.

JSP를 이용해서 만든 프로그램을 실행하려면 톰캣(Tomcat)이나 제티(Jetty) 또는 JBoss EAP와 같은 서버 프로그램이 필요하다. 단순 웹 서버가 정적인 HTML 파일이

나 이미지를 제공하는 것과 달리 이들 서버는 웹을 위한 연결, 프로그래밍 언어, 데이터베이스 연동과 같이 어플리케이션을 구현하는데 필요한 기능을 제공하고 있다. 이런 이유로 이들 서버 프로그램을 웹 어플리케이션 서버(Web Application Server), 줄여서 WAS라고도 부른다. 이들 WAS는 웹 브라우저로부터 요청이 오면 알맞은 프로그램을 찾아 실행하고, 프로그램의 실행 결과를 응답으로 전송한다.



[그림 2.8] WAS는 클라이언트의 요청이 오면, 알맞은 프로그램을 실행해서 응답을 생성한다.

WAS마다 지원하는 JSP 버전에 차이가 난다. 이 책은 JSP 2.3을 기준으로 하는데, 톰캣의 경우 톰캣 8 버전이 JSP 2.3 버전을 지원한다. 톰캣 7 버전은 JSP 2.2까지만 지원하기 때문에, JSP 2.3에서만 제공하는 기능을 사용하고 싶다면 톰캣 8 버전을 사용해야 한다.

JEE, JSP, 서블릿

자바 엔터프라이즈 에디션(Java Enterprise Edition), 줄여서 JEE는 자바를 이용해서 어플리케이션을 개발하는데 필요한 표준을 정의한 것으로서 JSP, 서블릿, JSTL, JPA 등 여러 표준으로 구성되어 있다. JEE 버전마다 구성된 기술들의 표준이 다른데, JEE 7 버전의 경우 JSP 2.3과 서블릿 3.1 버전을 사용하고 있다.

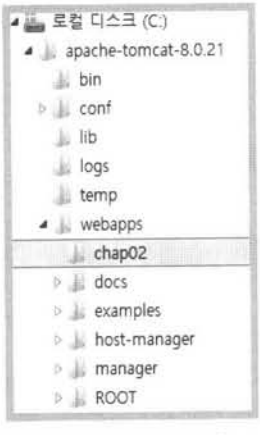
뒤에서 살펴보겠지만 JSP는 서블릿을 기반으로 동작하기 때문에, JSP 버전마다 해당하는 서블릿 버전이 정의되어 있다. 예를 들어, JSP 2.3 버전은 서블릿 3.1 버전을 기반으로 하고 있으며, JSP 2.2 버전은 서블릿 3.0 버전을 기반으로 하고 있다. 이런 이유로 JSP 2.3을 지원하는 톰캣 8 버전은 서블릿 3.1을 지원한다.

02 JSP 만들고 실행해보기

웹에 대한 기본 지식은 이 정도로 마치고, JSP로 간단한 웹 프로그램을 만들어보자. 여기서 만들 웹 프로그램은 현재 시간을 출력해주는 프로그램이다. 절차는 다음과 같다.

1. [톰캣폴더]\webapps\chap02 폴더를 만든다.
2. 편집기를 사용해서 chap02 폴더에 time.jsp 파일을 작성한다.
3. 톰캣을 실행한다.
4. 웹 브라우저에서 http://localhost:8080/chap02/time.jsp 주소를 입력한다.
5. 실행 결과를 확인한다.

먼저 [톰캣폴더]\webapps 폴더에 chap02 폴더를 생성한다. 톰캣을 C:\apache-tomcat-8.0.21에 설치했다면, C:\apache-tomcat-8.0.21\webapps\chap02 폴더를 생성하면 된다.



[그림 2.9] 예제 코드를 넣을 chap02 폴더 생성

톰캣에서 webapps 폴더가 무엇인지에 대한 내용은 4장에서 자세히 살펴본다. 일단 지금은 JSP를 비롯해서 웹과 관련된 코드를 webapps 폴더의 하위 폴더에 위치시켜야 한다는 정도로만 알고 넘어가자.

chap02 폴더를 생성했다면 webapps 폴더에 time.jsp 파일을 작성한다. 파일을 작성할 때에는 윈도우에 기본 내장된 메모장을 사용하거나 Notepad++나 Sublime과 같은 별도의 편집기를 설치해서 사용하면 된다. 편집기를 실행했다면 다음과 같은 [리스트 2.1]의 코드를 작성한 뒤 chap02 폴더에 time.jsp라는 이름으로 저장한다.

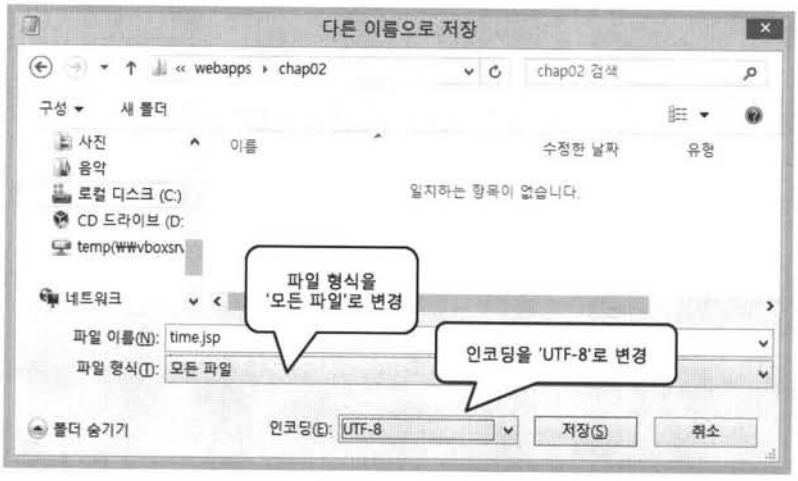
[리스트 2.1] chap02\time.jsp

```
01: <%@ page contentType="text/html; charset=UTF-8" %>
02: <html>
03: <head>
04: <title>현재 시간</title>
05: </head>
06: <body>
07: 지금 : <%= new java.util.Date() %>
08: </body>
09: </html>
```

NOTE

[리스트 2.1]의 왼쪽 편에 표시된 01, 02, 03 등의 숫자는 소스 코드의 행 번호를 표시한 것이므로 소스 코드를 작성할 때 입력하면 안 된다. 설명할 때의 편의를 위해 행 번호를 표시한 것이니 행 번호는 절대로 입력하지 않는다.

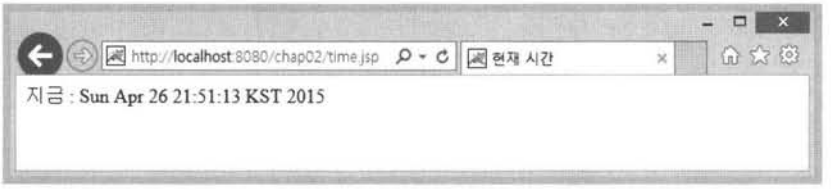
예제 코드를 저장할 때 주의할 점은 파일을 UTF-8 인코딩으로 저장해야 한다는 것이다. 메모장에서는 [그림 2.10]과 같이 인코딩 옵션을 'UTF-8'로 변경한 뒤에 저장하면 된다. 파일의 확장자가 jsp이므로 파일 형식도 '모든 파일'로 변경한 뒤에 저장한다.



[그림 2.10] 메모장에서 소스 코드를 UTF-8 인코딩으로 저장하기

소스 코드를 저장했다면 톰캣을 실행한다. 탐색기에서 [톰캣폴더]\bin\startup.bat 배치 파일을 더블클릭해서 톰캣을 실행하거나 명령 프롬프트에서 startup.bat 파일을 실행하면 된다.

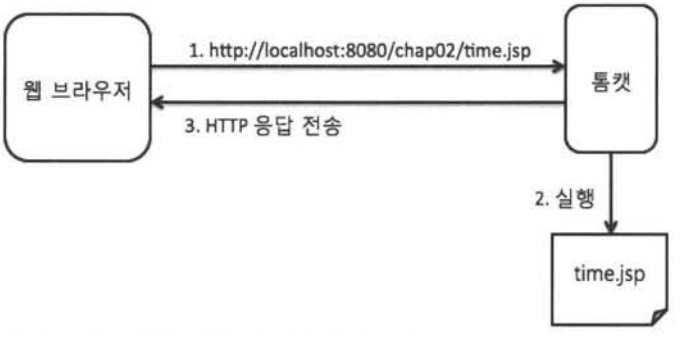
톰캣을 구동했다면 웹 브라우저에서 http://localhost:8080/chap02/time.jsp를 실행해보자. 그러면 [그림 2.11]과 같이 화면에 현재 시간이 표시되는 것을 확인할 수 있다.



[그림 2.11] time.jsp를 실행한 결과

에러 화면이 발생했다면 이 장의 뒷쪽에 있는 '몇 가지 에러' 내용을 읽어보고 정상적인 결과 화면을 볼 수 있도록 에러를 해결하고 넘어간다.

time.jsp를 다시 실행하거나 새로 고침을 하면 화면에 표시되는 시간 값이 변경된다. 즉, 실행할 때마다 결과가 다르므로 time.jsp는 동적 자원에 해당된다. 실행할 때마다 매번 다른 결과가 표시되는 것은 웹 브라우저에 주소를 입력할 때마다 [그림 2.12]와 같이 톰캣이 JSP를 매번 실행하기 때문이다.



[그림 2.12] 톰캣은 요청이 올 때마다 JSP를 실행한다.

time.jsp에서 시간을 출력해주는 부분은 07행의 다음 코드이다.

```
지금 : <%= new java.util.Date() %>
```

java.util.Date는 시간 값을 처리할 때 사용되는 자바 클래스다. <%=와 %>는 스크립트릿(scriptlet)이라는 JSP의 스크립트 요소로서 <%=와 %> 사이에 위치한 값을 문자열로 생성해준다. [그림 2.11]의 실행 결과에서는 다음과 같이 스크립트릿 영역이 문자열을 생성한 것이다.

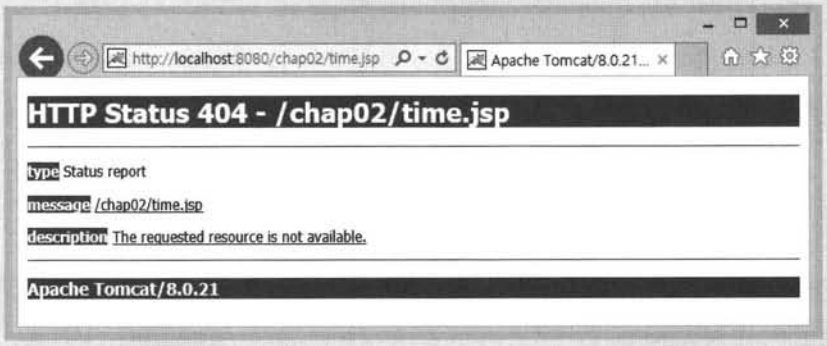
```
<%= new java.util.Date() %> ➡ Sun Apr 26 21:51:13 KST 2015
```

스크립트릿에 대한 내용은 3장에서 자세히 살펴보도록 하고, 실행할 때마다 JSP가 실행되고 스크립트릿을 이용해서 변경되는 부분을 생성한다는 정도로만 알고 넘어가자.

JSP 코드를 작성하고 웹 브라우저에서 실행해봤다. 웹 프로그래밍 자체는 이 장의 time.jsp처럼 어렵지 않다. 단지 JSP 코드를 작성하는데 필요한 문법과 코드 구성 방법을 익히면 된다. 3장부터 본격적으로 JSP에 대한 내용을 학습할 것이므로 집중해서 JSP를 익혀보도록 하자.

몇 가지 에러

time.jsp를 실행했는데 에러 화면을 만났다면 당황하지 말고 에러를 처리해보자. 먼저, JSP를 실행했는데 다음과 같은 에러 응답을 만날 수 있다.



에러 화면을 보면 'HTTP Status 404'라는 메시지를 볼 수 있는데, 이는 /chap02/time.jsp에 해당하는 JSP 파일이 없다는 것을 의미한다. 404 에러가 발생했다면 다음을 확인한다.

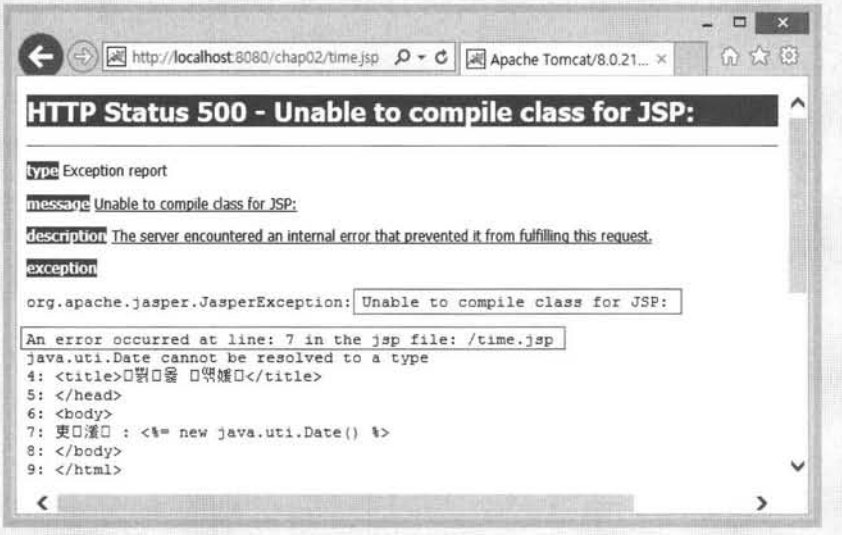
- [톰캣폴더]\webapps\chap02 폴더에 time.jsp 파일이 존재하는지
- 웹 브라우저에 입력한 주소가 http://localhost:8080/chap02/time.jsp가 맞는지

메모장에서 소스 코드를 작성하다보면 time.jsp가 아닌 time.jsp.txt와 같이 txt 파일로 저장하는 실수를 자주한다.

또 자주 만나는 에러는 500 에러이다. 처음 JSP 코드를 작성할 때 500 에러가 발생하는 경우는 대부분 소스 코드에 오타를 입력했기 때문이다. 예를 들어, [리스트 2.1]의 07행의 코드를 다음과 같이 입력했다고 해 보자.

```
지금 : <%= new java.uti.Date() %>
```

java.util.Date라고 입력해야 하는데, util에서 'l'자를 실수로 빼먹었다. 이 경우 다음과 같은 에러 화면이 표시된다.



소스 코드 자체에 오타가 있는 경우 'Unable to compile class for JSP'라는 메시지가 출력되며, 몇 번 행에 소스 코드 오류가 있는지 표시해준다. 위 에러 화면에서는 7번 행에 오류가 있다고 알려주고 있다. 위와 비슷한 에러 메시지가 표시되면 당황하지 말고 행 번호를 확인하고 잘못 입력한 코드는 없는지 눈을 크게 뜨고 살펴보도록 하자.