지식soft

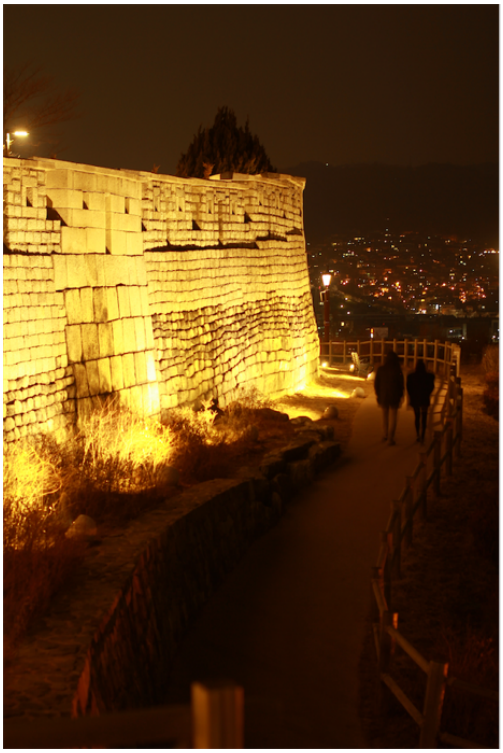
가볍지 않은 Web 프로그래밍

김세훈 지음

가볍지 않은
Web 프로그래밍

김세훈 지음

지식소프트(http://jisiksoft.com) 발행



표지 사진은 '지식소프트' 사무실 근처의 낙산공원 야경입니다.

가볍지 않은 Web 프로그래밍

초판발행 2016년 7월 25일

지은이 김세훈 / **펴낸이** 김세훈
펴낸곳 지식소프트 / **주소** 서울시 성북구 낙산길 243-15, 101-1004
전화 02-743-0367 / **팩스** 02-6280-0367
등록 2016년 6월 20일 제307-2016-38호
ISBN 979-11-958378-0-9 15000 / **정가** 25,000원
편집 남혜정 / **디자인** 김세훈

이 책에 대한 의견이나 오탈자 및 잘못된 내용에 대한 수정 정보는 아래 이메일로 알려주십시오.
이메일 thankyou@jisiksoft.com

Published by JISIKSOFT

저자 소개

지은이_ **김세훈**

2016 년을 맞이하며 확고히 한 원칙이 있다. "현재의 상황에서 최선의 선택을 한다." 우리는 선택하지 않고는 살아갈 수 없다. 밥 먹는 것 조차도 무엇을 먹을지 선택해야 한다. 현재의 선택이 나의 미래를 만들어 가는 것이고, 과거의 교훈은 현재의 선택을 도와준다. 내 인생은 내가 결정하는 것이라 20 대에 알았음에도 세상의 무서움에 힘겨워 하며 살아왔다. 그 무서움을 딛고 작년, 1 인 기업 '지식소프트'를 차렸다. '지식소프트' 에서 부장으로 평생 재직하며 먹고 싶은 것 먹으며 살고자 하나, 1 인 회사에서도 항상 선택해야 할 것들이 많아서 머리가 아프다.

저자 서문

1990년대 말부터 시작된 WWW(World Wide Web)의 인터넷 문화는 많은 사람들이 컴퓨터를 사용하는데 일조하였으며, 방대한 정보의 공유를 만들어 주었다. 시대가 발전하여 현재는 컴퓨터 없이는 살아도 스마트폰 없이는 생활이 안 되는 시대가 되었으나 컴퓨터건 스마트폰이건 인터넷의 정보를 보기 위해서는 브라우저가 필요하다.

이 책에서는 브라우저에서 정보를 보여 주기 위한 웹 프로그래밍을 다룬다. 현재는 HTML 언어를 사용해서 웹 페이지를 만들고, HTML 코드를 해독하는 브라우저로 인터넷 세상의 정보를 접한다. HTML 언어로 웹 페이지를 만드는 작업은 간단하기 때문에 예전에는 웹 프로그래밍을 하는 사람들의 기술을 높게 생각하지 않았다.

브라우저에서 보여지는 화면을 만드는 HTML, CSS, JavaScript 언어들만 사용하는 프로그래머 보다는 서버에서 움직이는 JSP 또는 PHP 프로그램을 만드는 사람들을 더 고급 기술자로 생각했다. 물론, 현재 인터넷에서 서비스하는 많은 홈페이지들의 기술의 난이도는 높지가 않은데, JavaScript 언어로 프로그램을 만들어도 이벤트를 처리하는 단순 코드작업이 많은 편이다. 간단히 표현하면, 브라우저에서 진행되는 JavaScript 코드는 간단한 기능을 수행하고, 서버에서 동작하는 프로그램들이 데이터베이스와 연동하면서 복잡한 기능들을 수행하는 구조다. 필자도 JavaScript 언어를 깊이 있게 이해하지 못했을 때는 C/C++, Java 언어보다 훨씬 낮은 수준의 프로그램 언어로 생각했었다. 그러나, JavaScript 언어로 클래스를 만들어서 프로그래밍을 할 수 있게 되자 JavaScript 언어에 매료되었고, JavaScript 언어가 다른 프로그래밍 언어에서 제공하는 대부분의 연산자를 제공하고 있기 때문에 프로그래밍을 하는데 제약이 많지 않다. 특정 서비스를 제공하는 시스템을 만들 때 서버에서 동작하는 프로그램도 중요하지만, JavaScript 언어를 깊이 있게 다루는 고급 프로그래머가 점차적으로 많이 늘어날 수밖에 없는 이유다.

특히, 앞으로는 브라우저에서 실행하는 온라인 게임들이 많이 만들어지고 Flash 같은 프로그램을 사용하지 않고, JavaScript 언어로 난이도가 높은 프로그램을 만드는 웹 프로그래머들을 시장에서 많이 요구할 것이다.

웹 프로그래밍Web Programming은 이전의 책에서 소개한 MFC와 같이 그래픽Graphic 프로그래밍에 속한다. 사용자가 보기 편하고 화려한 화면을 가진 프로그램에서 생각해야 할 부분은 UIUser Interface 의 속성Property과 이벤트Event 두 가지를 이해할 필요가 있다. UI는 화면에 다양한 그림을 그려주고 위치, 크기, 색과 같은 보기 좋은 모양을 만들어 주는 것이며, 이벤트는 사용자가 마우스 버튼을 누르는 등의 동작인데, 이벤트가 발생했을 때 화면을 어떻게 처리해야 하는지를 프로그래밍하게 되는 것이다.

HTML과 **CSS**가 화면의 내용과 멋지게 보여지기 위한 디자인 속성의 영역이라면, **JavaScript**는 이벤트에 대한 프로그래밍을 하는 영역이다. 즉, 인터넷을 하는 사용자가 보는 브라우저에서의 프로그래밍은 HTML, CSS, JavaScript를 가지고 프로그래밍을 만드는 것이다. JavaScript로 프로그래밍을 쉽게 하기 위해서 JavaScript 함수를 만들어 놓은 라이브러리들이 많이 있는데, 이 중에서 가장 많이 사용하는 것이 jQuery다. jQuery를 사용하지 않고 JavaScript만으로도 프로그램을 만들 수 있지만, jQuery를 사용하면 다양한 기능들을 쉽게 사용할 수 있기 때문에 개발 시간이 절약된다. 이러한 원리로 그래프를 그려주는 함수들을 만들어서 라이브러리를 제공하기도 하기 때문에 개발하고자 하는 웹 프로그래밍에 맞는 라이브러리를 찾아서 보기 좋은 화면을 만드는 것은 웹 프로그래밍에서 중요하다. 간혹, 라이센스 비용을 지불해야 하는 라이브러리도 있지만, 직접 만들었을 때 소비되는 개발 시간을 생각하면 그다지 큰 비용이 아닐 수도 있다.

사용자는 브라우저를 통해 인터넷 정보를 보게 되지만, 사용자에게 정보를 주는 곳은 인터넷에 연결된 서버다. 서버는 브라우저를 통해 정보를 보는 곳이 아니라, 필요한 정보를 전달해 주는 곳이기 때문에 Java나 C/C++과 같은 프로그래밍 언어로 개발하는데, 웹 서버에서 프로그램을 구현하기 위하여 만들어진 것이 **JSP**와 **PHP**라고 이해하면 된다. JSP Java Server Pages는 Java 계열의 프로그래밍 언어이고, PHP는 새로운 언어일 수 있지만 C/C++ 사용자가 좀더 친숙해 하는 프로그래밍 언어다. JSP는 대형 프로젝트에서 사용을 많이 하고, 규모가 작은 사이트에서는 PHP를 사용한다. JSP와 PHP를 간단히 비교하면, 컴파일을 해서 실행파일로 관리하는 JSP는 컴파일 과정없이 코드를 분석해서 실행하는 스크립트 언어인 PHP보다 훨씬 빠르기 때문에 사용자가 많이 접속하는 대형 사이트에서는 JSP를 사용해서 시스템을 구축한다.

이 책에서는 웹 프로그래밍에 대한 전반적인 이해를 위해서 사용하기 쉬운 PHP 언어로 서버의 동작원리를 설명한다. 기본 언어를 소개하는 첫 번째 Chapter는 각 장

이 연결되어서 설명하는데, PHP 언어를 설명하는 1.5장의 내용은 오히려 JavaScript의 고급 기술을 더 많이 설명한다. JSP를 사용하는 독자여도 PHP의 내용을 깊이있게 보기를 권한다.

이 책에서는 가장 기본적인 정보를 보여 주는 방법을 간략히 설명한 후 단계적으로 좀더 고급스러운 웹 페이지를 만드는 방법을 설명하는데, 실생활에서 쉽게 사용할 수 있는 설문지 만드는 방법과 저자가 현재 몸담고 있는 1인 회사인 '지식소프트' 홈페이지를 만드는 방법 등을 설명한다. 독자가 좀더 쉽게 이해하는 방법은 책과 함께 제공되는 소스 코드Source Code를 먼저 실행해 보는 것이다. 이 책은 "Chapter 1"에서 웹 프로그래밍 언어의 기본을 설명하고, 해당 언어들의 특징을 실제 코드로 이해하는 방법으로 진행한다.

이 책을 통하여 웹 프로그래밍은 많은 것들을 모두 알고 만드는 것이 아니라, 기본적인 몇 가지를 깊이 이해하면 된다는 것을 알게 될 것이다. 필자가 기획한 'How-to Series'의 4번째 책인 '웹 프로그래밍'은 실제 프로그램을 만들면서 이해하는 과정이며, 이 책의 내용을 응용하면 자신이 원하는 웹 프로그램을 쉽게 만들수 있을 것이다. 필자의 프로그래밍하는 기본 방식은 "외우지 않고 필요한 것들을 인터넷에서 찾아서 응용하여 조립한다."인데, 필자처럼 기억력이 나빠도 인터넷에는 웹 프로그래밍에 관련한 예제와 정보들이 많기에 밥 먹고 살 수 있는 것 같아 감사하다. 프로그래밍은 반복하다 보면 자연적으로 익숙해 지는 분야라서 암기력이 덜 필요하다. 그래서 저자가 진행하고 있는 How-to Series는 방법론적인 부분들에 초점을 맞추어 집필했다. 매년 하나의 주제를 가진 프로그래밍 책을 한 권씩 만들고 있다. 일방적인 지식의 전달 대신 "이러한 방법도 있구나"하고 이해하고 응용하는 것이 중요하다.

How-to Series 네 번째 책 Web Programming을 현명하지 못한 저자와 함께하는 사랑하는 아내 남혜정에게 바친다. 또한, 친구 권무준의 예쁜 딸 권규민에게 이 책을 선물로 준다.

PS : 예제 파일은 다음에서 다운로드할 수 있습니다.
https://github.com/jisiksoft/web_programming

차례

chapter 1

웹 프로그래밍 기본 언어 소개

프로그래밍 언어를 이해하기 위해 먼저 기본적인 골격부터 살펴보자. 마우스가 없던 시절에는 문자Text만으로 컴퓨터 프로그래밍을 했기 때문에 논리적인 훈련이 많이 필요했지만 현재는 멋진 화면을 보면서 마우스만으로도 프로그램을 동작시켜야 하는 시대가 되었다. 그래서 화려한 화면을 만들고, 사용자가 쉽게 운영할 수 있도록 구현해야 한다. Windows, Mac, Linux 같은 운영체제에서 운영하는 화려한 프로그램을 만들 수도 있으나, 최대한 많은 사람들이 사용하는 프로그램을 만드는 방법 중의 하나는 특정 운영체제에 한정되지 않은 웹 프로그래밍Web Programming을 만드는 것이 좋다. 초창기 웹 프로그래밍이 문자Text와 사진Picture만을 가지고 정보를 전달하는 것이 목적이었다면, 현재는 하드웨어의 발전과 함께 브라우저에서 동적으로 움직이는 웹 프로그래밍으로 발전하고 있으며, 앞으로는 게임들도 브라우저에서 즐길 수 있게 점차 변해갈 것이다.

웹 프로그래밍은 사용자가 사용하는 브라우저에서 동작하는 코드를 만들고, 코드를 전달하는 서버Server 프로그래밍도 익혀야 한다. 인터넷에는 많은 웹 프로그래밍 예제들이 있는데, 이 책에서 다루는 것은 처음 입문하는 사람들에게 원리를 이해시키는 것에 중점을 두고 코드를 구현한다. 필자가 만드는 How-to Series의 기본은 우리에게 필요한 프로그램을 만들면서 프로그래밍을 함께 공부하며, 이후에는 독자 스스로 원하는 프로그램을 직접 만들수 있도록 하는 것이다. 웹 프로그래밍에서 사용하는 모든 언어의 내용을 자세히 기술한 책이어도 설명이 많은 것일 뿐 책의 모든 내용을 세세히 기억할 수 없다. 그러나, 본 책을 이해한 독자라면 웹 프로그래밍에 대한 전반적인 개념을 이해한 토대 위에 재미있는 웹 프로그램을 만들 수 있을 것이다.

브라우저에서 사용하는 프로그래밍 언어는 크게 HTML, CSS, JavaScript이다. CSS는 화면을 이쁘게 보이기 위한 디자인 속성을 나타내기에 프로그래밍 언어로 보지 않는 사람도 있지만, 디자인이란 것은 프로그램의 가치를 높여주는 상품의 포장지이기

때문에 CSS의 가치는 높다고 본인은 평가한다. HTML(HyperText Markup Language)과 CSS(Cascading Style Sheets)는 브라우저에서 보여지는 화면의 내용과 디자인 속성을 위한 것이며, JavaScript는 사용자의 동작에 반응하여 이후에 처리되는 과정을 프로그래밍으로 처리하는 것이다. 프로그래밍을 잘하는 프로그래머는 JavaScript의 비중을 높이 생각하는 편이지만, HTML과 CSS를 이해하지 못하면 멋진 웹 프로그래밍을 한다고 할 수 없기 때문에 모든 언어에 대해 깊이 있게 이해할 필요가 있다.

그림 1-1 웹 프로그래밍의 기본 언어 (3 가지)

[그림 1-1]은 웹 프로그래밍의 세가지 기본 언어들을 보여주는데 HTML은 화면에 보여지는 글자 및 그림을 보여주기 위한 내용을 위한 것이고 CSS는 HTML의 디자인 속성을 담당한다. 또한 JavaScript는 브라우저에서 사용자가 마우스 버튼 등을 누를 때 발생하는 이벤트(Event)등에 대한 처리를 위한 프로그래밍 코드로 이해하면 된다. 웹 프로그래밍의 골격을 이와 같이 생각하면 이후에 진행되는 각각의 프로그래밍 언어에 대한 이해가 쉽다.

이번 Chapter에서는 HTML, CSS, 그리고 JavaScript로 시작하지만, 웹 프로그래밍에서 가장 많이 사용하는 것중의 하나인 jQuery 사용법과 브라우저에 정보를 보내주는 서버에서 동작하는 PHP도 간단히 설명하여 이후에 진행되는 웹 프로그래밍을 쉽게 이해할 수 있도록 했다. 5장으로 이루어진 이번 Chapter는 개별 주제로 나누었지만, 1장부터 차례대로 연결되어 이해하기 쉬운 구조를 만들었다. 또한, 이전 장에서 간단히 다룬 개별 주제의 심도 있는 내용은 다음 장에서 다루는 언어에서 좀더 깊이 있게 추가 설명하였다.

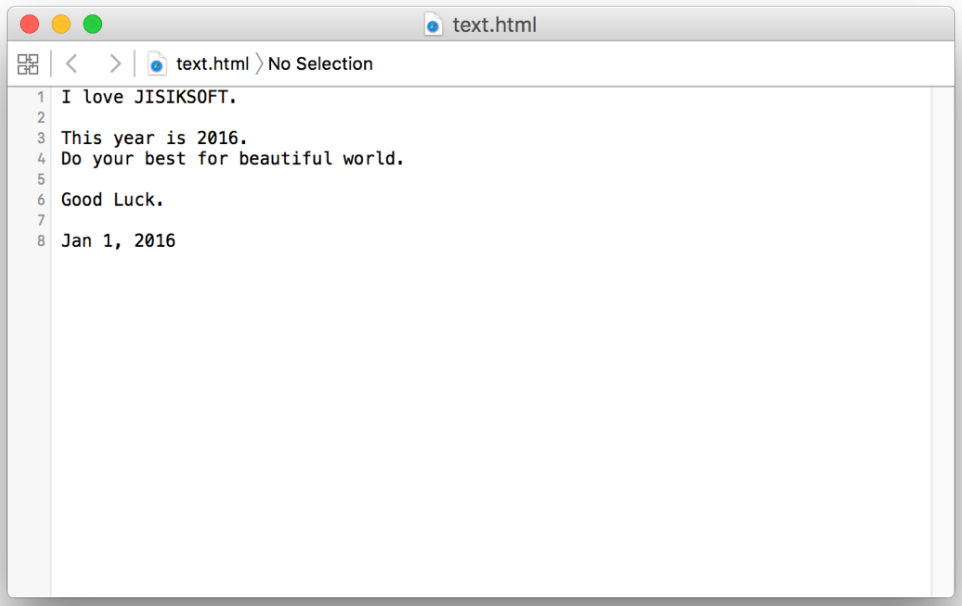
현재 대형 사이트는 많은 사람들의 협업에 의해서 만들어지기 때문에 혼자서 간단한 시스템 전체를 만들 수 있는 것이 불가능하다고 생각하는 사람들이 의외로 많다. 그러나, 전반적인 개념을 이해한 독자라면 시간을 가지고 하나씩 구현하면서 웹 프로그래밍의 깊이를 느낄 수 있을 것이다. 프로그래밍이란 것은 한 타 한 타 타이핑을 하면서 한 줄을 만들고, 그 한 줄들이 모여서 몇천 몇만 줄이 되면서 멋진

소프트웨어를 만드는 작업이다. 원리를 이해하면 게임보다 즐거운 프로그래밍의 세계에 빠져들 수 있을 것이다. "How-to Series"의 개념은 모든 프로그래밍의 소스는 인터넷에 공개되어 있으며, 프로그래머는 그것들을 잘 조합해서 멋진 소프트웨어를 만드는 것이다. 물론, "Copy & Paste"가 아닌, 나만의 알고리즘으로 재생산하는 것이다.

1.1 HTML

HTML의 코드를 작성하기 위해서는 Windows의 메모장과 같은 에디터Editor를 사용하면 된다. 하지만 프로그래밍을 하기 위하여 일반적으로 사용하는 이클립스Eclipse 나 애플Apple 맥Mac에서 사용하는 Xcode 같은 프로그래밍 에디터Editor를 사용하는 것이 좋다. HTML 파일은 확장자가 .html로 만들어진 것이며, html 파일을 실행시키면 기본 브라우저가 실행되어 파일의 내용을 해당 브라우저에서 볼 수 있다. 즉, 글을 입력하고 .html 확장자로 파일을 만들면 브라우저에서 실행되는 HTML 파일이 만들어지는 것이다. [그림 1-2]는 에디터에서 text.html 파일을 만든 것이며 [그림 1-3]은 이것을 브라우저에서 실행시킨 결과이다.

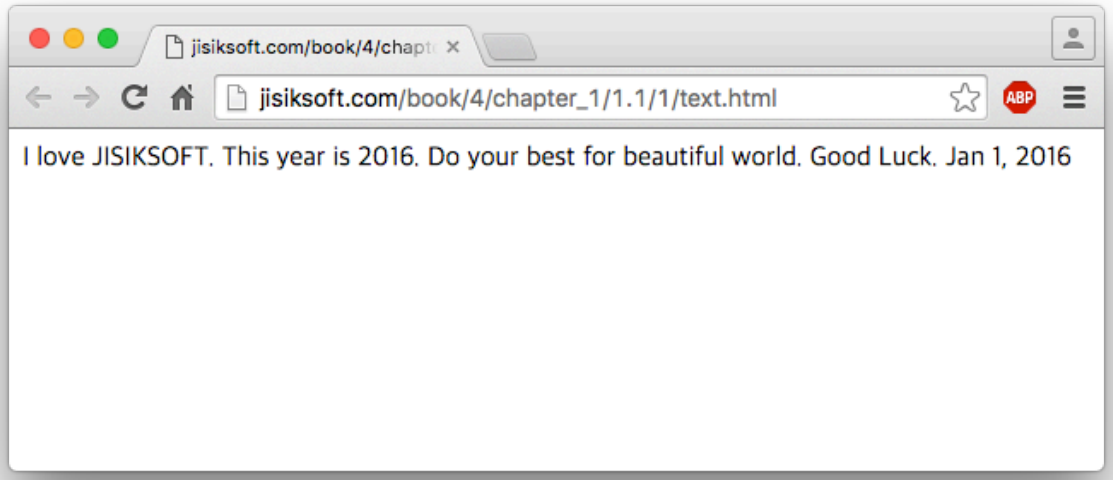
그림 1-2 텍스트 Text 만 넣은 html 파일 (/1/text.html)



이 책은 웹 프로그래밍이기 때문에 인터넷이 연결된 모바일에서도 실행결과를 확인할 수 있게 했는데, 지식소프트의 서버에 있는 코드를 아래의 링크Link를 통하여 확인이 가능하다. 이 책에서 제공하는 소스 코드를 다운 받아서 text.html 파일을 더블 클릭하면 독자의 컴퓨터에 있는 기본 브라우저가 실행되면서 같은 결과를 얻을 수 있다. 브라우저에서 동작하는 HTML, CSS, JavaScript 코드는 사용자 컴퓨터에서 직접 실행해 볼 수 있기 때문에 코드를 구현하는 것이 쉽다.

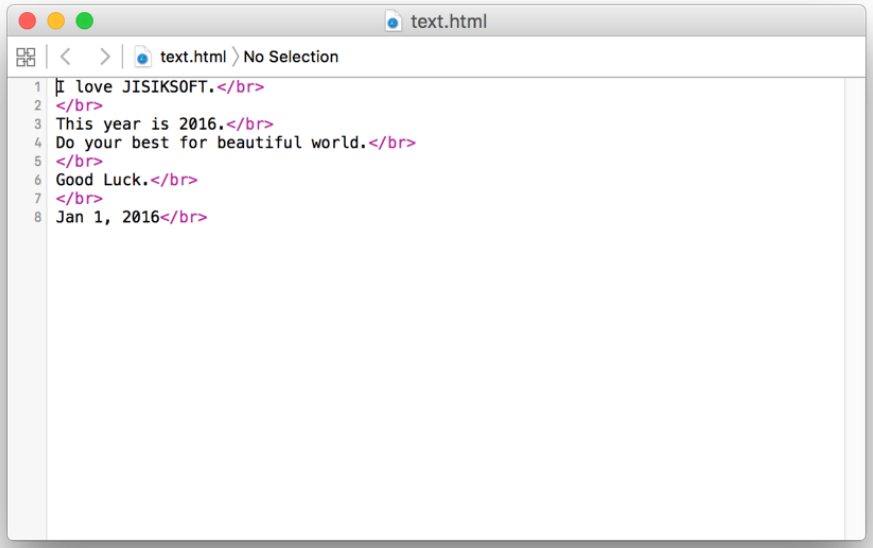
확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/1/text.html

그림 1-3 'text.html'을 실행한 후 브라우저에서의 결과 확인



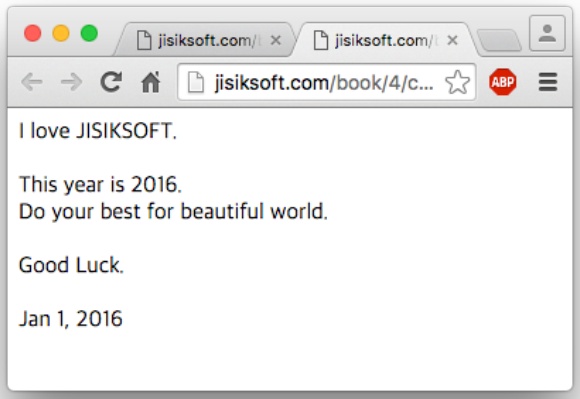
[그림 1-3]에서의 실행 결과는 test.html 파일에 있는 내용이 한 줄에서 표현된다. "How-to Series"의 세 번째 책("OCR 프로그래밍" p101 "3.2 줄바꿈 문자 추가하기")에서 설명한 줄바꿈 문자를 HTML에서도 추가를 해주어야 하는데, XMLExtensible Markup Language 언어의 한 종류인 HTML에서는 '
'를 추가하여 줄바꿈을 만들 수 있다. 일반적으로 XML 언어를 "마크업 언어"라고 부르는데 간단히 설명하며 '<'문자로 시작해서 '>'로 끝나는 규칙의 언어라고 생각하면 되는데, '
'은 간단한 마크업 언어이며 "<html>내용</html>" 등의 같은 표현으로 대부분 사용한다. ('<html>'을 시작 태그Tag라고 하며, '</html>'을 끝 태그라고 부른다.) [그림 1-4]는 '
' 태그를 사용하여 브라우저에서 줄바꿈을 하도록 내용을 만든 것이며, [그림 1-5]에서와 같은 결과를 얻게 된다.

그림 1-4 줄바꿈 태그 Tag </br>을 추가한 html 파일 (/2/text.html)



확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/2/text.html

그림 1-5 줄바꿈 태그 </br>이 추가된 'text.html'을 실행한 결과 확인



HTML 언어는 기본 골격을 가지고 구현이 되어야 하는데, 이전과 같이 텍스트만 작성하여도 되지만, HTML이라고 하면 기본적으로 [그림 1-6]과 같은 구조로 작성되어야 한다. <html> 태그로 시작해서 </html> 태그로 종료되어야 하며, 이 태그 안에 내용이 들어가야 한다. <html></html> 태그 안에는 <head></head> 태그와 <body></body> 태그가 존재하는데, <head></head> 태그 안에는 HTML 문서의 설정 등이 정의되고 <body></body> 태그 안에는 브라우저에 출력되는 내용을 넣으면 된다.

그림 1-6 HTML 파일의 기본 구조

```
<html>
    <head>
        html 문서의 설정등이 정의된다.
    </head>
    <body>
        브라우저에서 보여지는 내용이 기록된다.
    </body>
</html>
```

[그림 1-7]은 HTML 문서의 구조 안에서 [그림 1-4]의 텍스트를 표현하는데 보여지는 결과는 같지만 앞으로 추가구현을 통한 확장을 위해서는 이와 같이 만들어져야 한다. 태그를 가지고 만들어진 HTML 문서는 나중에 보게 되는 CSS를 사용하여 태그에 속성을 넣어서 브라우저에서 보기 좋은 디자인을 연출할 수 있다. [그림 1-7]의 start.html 파일에서 가장 위에 기입하는 태그는 '<!DOCTYPE html>' 인데 이것은 문서 형식Document Type의 XML 파일로서 HTML을 표현한다는 의미이며 현재는 가장 최신 버전인 HTML5 문서를 의미하지만, 모든 HTML 파일에 처음에 '<!DOCTYPE html>'을 사용한다고 생각하면 된다. <head></head> 태그 안에는 <meta></meta> 태그와 <title></title> 태그가 들어가는데, 두 가지를 넣는 순서는 중요하지 않다.

<meta></meta> 태그에는 문서의 요약Description, 인터넷에서 검색되는 핵심어Keyword, 작성자Author와 어느 형태의 문자 형식Character Set인지를 나타내는 정보 등을 넣게 되는데 예제에서는 문서의 문자 형식(charset)을 정의했다. 문자 형식Character Set으로서 'utf-8'을 정의하였는데, 한글을 표현하기 위해서 현재 가장 많이 사용하는 문자 인코딩 방식이며 이것을 정의하지 않으면 한글이 브라우저에서 깨져서 출력되는 경우가 많다. 간혹 영어가 아닌 다른 나라의 언어로 만들어진 인터넷 사이트의 문자가 깨져서 나오는 것은 이러한 문자 형식이 제대로 정의되지 않아서 생기는 문제가 많다.

<title></title> 태그에는 해당 HTML 파일의 제목을 넣게 되는데, [그림 1-8]을 보게 되면 브라우저 창의 탭에 적힌 글자가 "Hello JISIKSOFT"와 같이 <title></title> 태그 안의 내용과 같음을 알 수 있다.

<body></body> 태그 안에 있는 내용은 브라우저에 출력하는 내용으로 [그림 1-4]와 같은 내용이다. 실제 코드에서 <meta> 태그와 같은 다수의 태그들은 종료

태그(ex: </meta> 또는 </img>)를 생략해도 되지만, 필자는 모든 태그는 종료 태그를 사용하는 것이 좀더 엄격한 XML 언어에 부합된다고 생각해서 항상 종료 태그를 사용한다.

그림 1-7 HTML 기본 골격 코드 (/3/start.html)

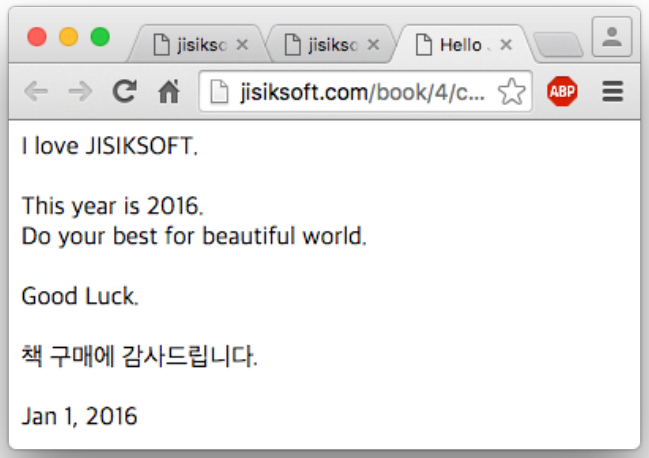
start.html

start.html 〉 No Selection

```
<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8"></meta>
        <title>Hello JISIKSOFT</title>
    </head>
    <body>
        I love JISIKSOFT.</br>
        </br>
        This year is 2016.</br>
        Do your best for beautiful world.</br>
        </br>
        Good Luck.</br>
        </br>
        책 구매에 감사드립니다.</br>
        </br>
        Jan 1, 2016</br>
    </body>
</html>

```

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/3/start.html

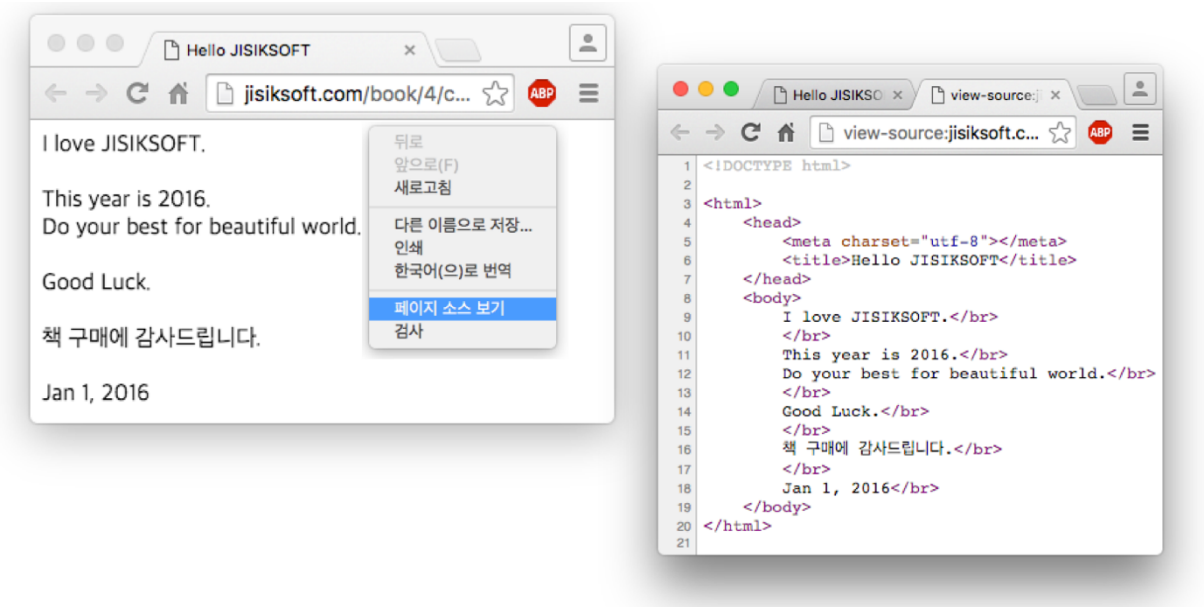
그림 1-8 'start.html' 파일을 실행한 결과



[그림 1-9]는 브라우저에서 마우스 오른쪽 버튼을 눌러서 "페이지 소스보기"를

실행한 화면이며 HTML 문서가 그대로 보인다. 이러한 이유로 누군가 HTML 문서를 만들어도 다른 사람들이 브라우저를 통하여 HTML 문서를 쉽게 복사할 수 있는데, 이러한 이유로 HTML 언어의 개념만 이해하면 쉽게 만들수 있다는 인식 때문에 HTML 언어를 가볍게 보곤 한다. 누군가 멋진 인터넷 사이트를 만들어도 이미지만을 변경해서 쉽게 비슷한 사이트를 만들 수 있으며, 인터넷 사이트의 특성상 이미지만 변경해도 사용자들은 전혀 다른 사이트라고 생각할 수 있지만, 프로그래머의 관점에서 보면 같은 기술이라고 판단한다. 홈페이지를 만드는 소규모의 회사들이 의외로 많은 편인데, 하나의 인터넷 사이트만 만들어 놓으면 다른 사이트는 디자이너가 이미지를 새롭게 만들어서 별 개의 사이트로 만드는 것이 용이하기 때문이다.

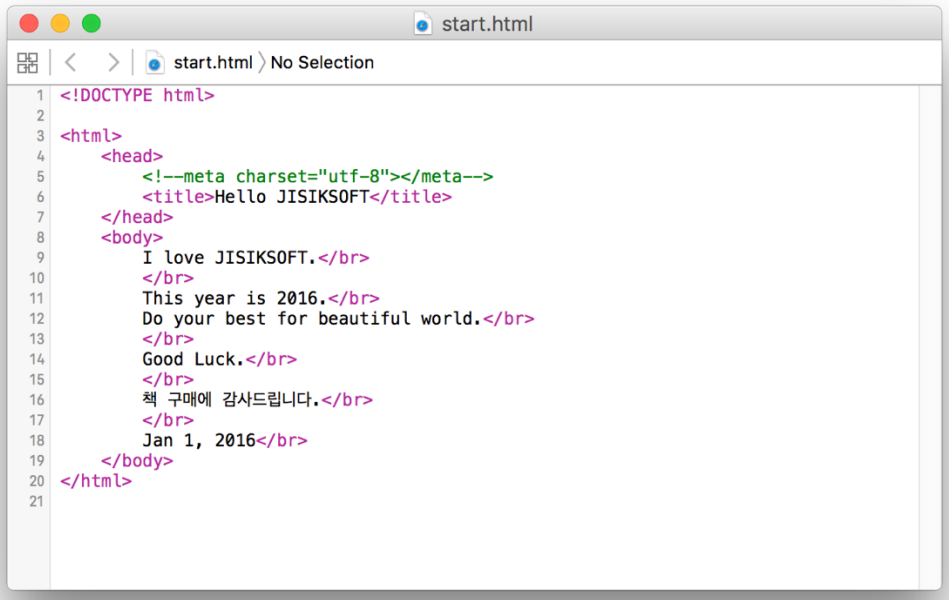
그림 1-9 "페이지 소스보기"로 문서를 확인한 결과



프로그램을 구현할 때 주석 처리를 하는 경우가 많은데, HTML 언어에서는 주석처리를 "<!-- 주석 -->"를 사용한다. [그림 1-10]에서 <meta></meta> 태그를 주석처리 하였는데, 일반적으로 코드를 작성 중에 해당 태그 코드가 필요하지 않다고 바로 지우는 것보다 주석처리를 사용하는 것이 좋을 때가 많다. 왜냐하면 나중에 해당 부분이 필요할 수도 있고 테스트를 위해서 잠시 주석을 해주는 경우도 많기 때문이다. [그림 1-10]에서는 주석으로 'utf-8'의 설정을 실행 코드에서 없애주었으며, [그림 1-11]에서와 같이 한글 부분이 정상적으로 출력되지 않는 결과가 발생했다. 모든 브라우저 환경에서 이렇게 한글이 깨지는 결과가 나타나는 것은 아니지만, 문자 형식Character Set의 정의가 중요함을 알 수 있다. 한글을 사용하지

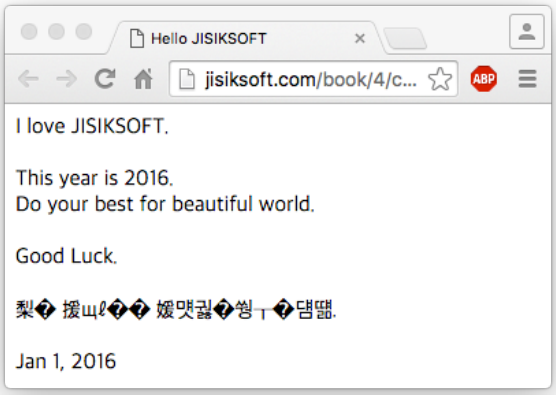
않는 외국인 사용자의 경우는 브라우저에서 한글이 보이도록 설정하지 않은 경우가 많기 때문에 대부분이 [그림 1-11]과 같이 나올 것이며, 'utf-8'을 설정한 경우에는 한글이 정상적으로 출력된다.

그림 1-10 한글을 위한 'utf-8'을 주석처리한 코드 (/4/start.html)



확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/4/start.html

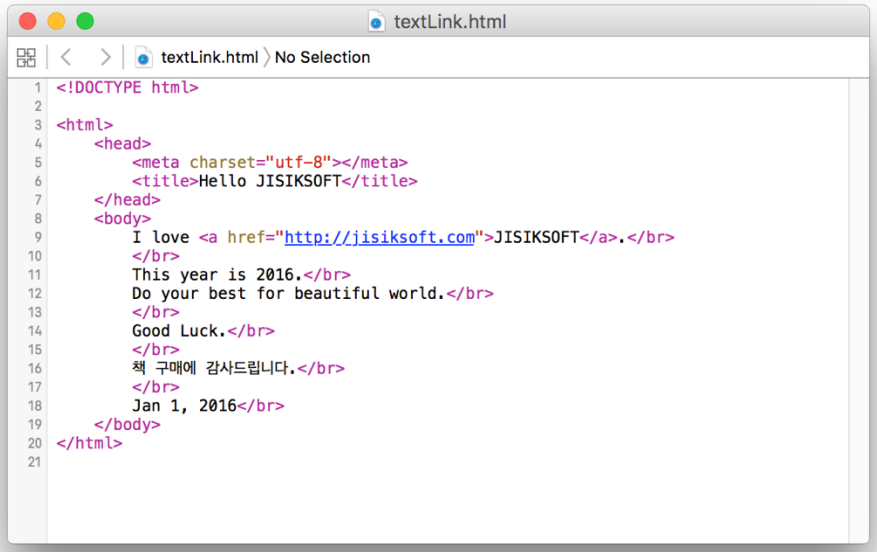
그림 1-11 'utf-8'이 설정되지 않으면 한글이 올바르게 출력되지 않는 브라우저가 있다.



인터넷 세상에서는 글자를 클릭하면 다른 페이지로 이동한다. 대표적인 것이 Google 사이트에서 검색을 한 후에 검색된 글자를 클릭하면 해당 페이지로 이동하는 것이다. 이것은 <a> 태그를 사용하여 구현하는데 [그림 1-12]에서는

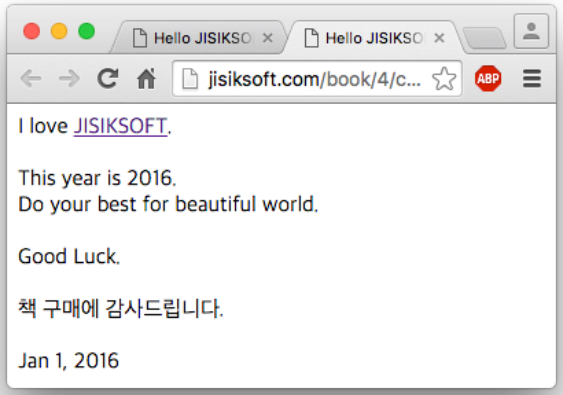
<a> 태그 안에서 다른 곳으로 링크Link시키는 속성인 href를 사용하여 JISIKSOFT 글자를 클릭하면 'http://jisiksoft.com'으로 이동하는 코드를 구현한 것이다. [그림 1-13]에서는 실행한 화면을 보여주는데, JISIKSOFT 글자의 색이 변하고 밑줄이 생겼으며 마우스를 글자 위로 옮기면 마우스의 이미지가 클릭할 수 있는 이미지로 변한다. 마우스로 JISIKSOFT 글자를 클릭하면 href로 정의된 지식소프트 홈페이지('http://jisiksoft.com')로 이동한다.

그림 1-12 마우스로 글자를 클릭시 다른 페이지로 이동하는 코드 (/5/textLink.html)



확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/5/textLink.html

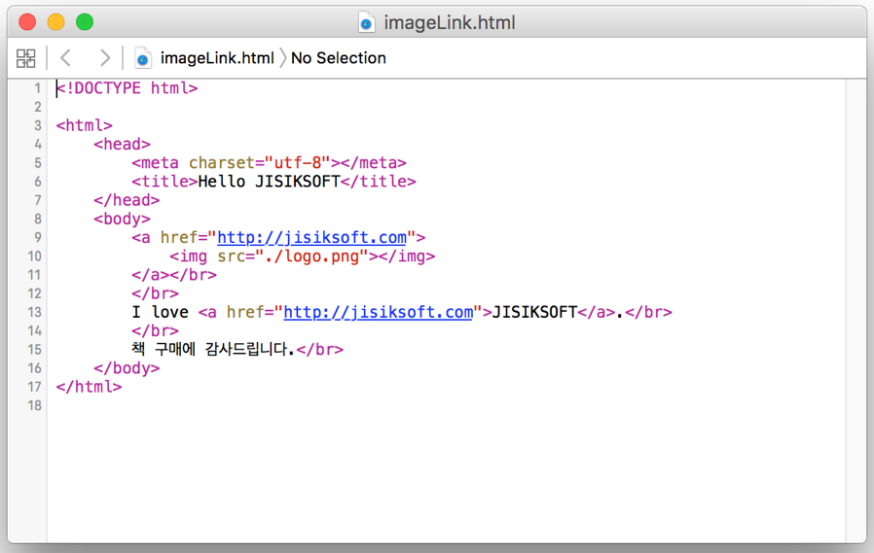
그림 1-13 브라우저에서 'JISIKSOFT' 글자를 클릭하면 'http://jisiksoft.com'으로 이동한다.



<a> 태그는 글자뿐만 아니라 모든 환경에서 사용할 수 있는데, [그림 1-14]는

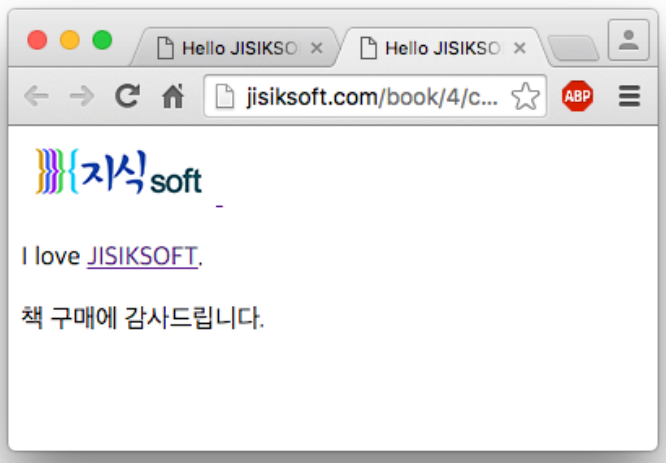
이미지를 화면에서 보여주는 <img> 태그를 <a> 태그로 감싸서 이미지를 클릭하면 정의된 페이지인 홈페이지로 이동하는 코드를 보여준다. <img> 태그는 다운 받는 이미지를 src(Source) 속성을 사용하여 파일 이름을 넣어주며, './'는 현재 디렉토리Directory에 있는 파일을 가져오라는 것이며, '../'는 이전 디렉토리로 이동하라는 의미이다. 컴퓨터를 사용하면 '.'와 '..'의 사용을 많이 보는데, 상대적인 파일 시스템의 접근을 의미한다. [그림 1-15]에서 로고를 클릭하거나 JISIKSOFT 글자를 클릭하면 홈페이지('http://jisiksoft.com')로 이동한다.

그림 1-14 마우스로 이미지를 클릭시 다른 페이지로 이동하는 코드 (/6/imageLink.html)



확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/6/imageLink.html

그림 1-15 브라우저에서 이미지를 클릭하면 'http://jisiksoft.com'으로 이동한다.



HTML에서는 테이블Table을 많이 사용하는데, 문서 작업 할 때 사용하는 '표'의 영어 이름이 테이블Table이다.

[그림 1-16]은 일반적인 문서에서 삽입하는 표인데, 브라우저에서도 쉽게 만들수 있다. 이는 수학에서 사용하는 행렬Matrix의 개념이다. 행렬에는 행Row과 열Column이 있으며 행과 열의 조합으로 테이블Table이 만들어 진다.

[그림 1-17]은 브라우저에서 표를 출력하기 위하여 사용하는 <table></table> 태그의 구조를 보여주는데, <table></table> 태그 안에는 행Row을 구분하기 위한 <tr></tr> 태그가 있으며, 행 안에서 열Column을 나누기 위하여 <td></td> 태그를 사용한다. 많이 사용하지는 않지만 표의 제목Head을 구분하기 위하여 <th></th> 태그의 사용도 가능하다. 실제 문서에서 각각의 위치에 있는 장소를 셀Cell이라고 하는데, 셀의 병합 등을 이용하여 테이블을 다양하게 만들 수 있다. 인터넷 검색을 하게 되면 쉽게 알수 있는 내용이라서 이 책에서는 가장 많이 사용하는 내용만 간단히 설명한다.

[그림 1-16]과 [그림 1-17]을 비교하면 표의 위치와 <table></table> 태그 위치의 확인이 가능하다.

그림 1-16 워드나 파워포인트에서 문서에 삽입하는 표

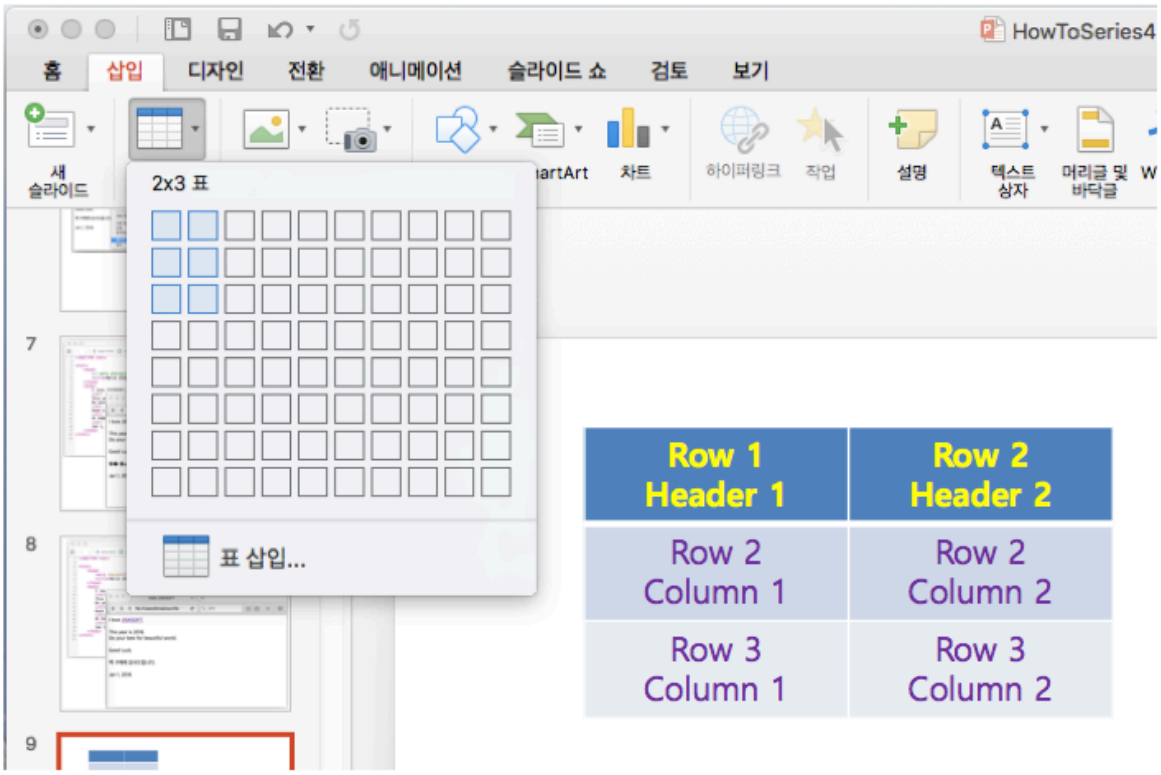


그림 1-17 Table 태그 구조

```
<table>
    <tr>
        <th> 행(Row 1) 제목(Header 1) </th>
        <th> 행(Row 1) 제목(Header 2) </th>
    </tr>
    <tr>
        <td> 행(Row 2) 열(Column 1) 내용 </td>
        <td> 행(Row 2) 열(Column 2) 내용 </td>
    </tr>
    <tr>
        <td> 행(Row 3) 열(Column 1) 내용 </td>
        <td> 행(Row 3) 열(Column 2) 내용 </td>
    </tr>
</table>
```

[코드 1-1]에서는 <table></table> 태그를 사용하여 지식소프트에서 발간한 서적 목록을 보여주는 코드를 작성하였다. 화면에서 보기좋게 만들기 위해서는 다음 장에서 설명하는 CSS의 디자인 속성이 필요하지만, 이번 장에서는 디자인 속성을 배제한 HTML 언어를 설명하기 때문에 테두리가 없는 표를 만든다.

[그림 1-18]에서는 코드의 결과인데 행과 열에 맞게 책의 목차가 보인다.

[코드 1-1] Table 만들기 (/7/table.html)

```
<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8"></meta>
        <title>Hello JISIKSOFT</title>
    </head>
    <body>
        <a href="http://jisiksoft.com">
            <img src="./logo.png"></img>
        </a></br>
        </br>
        <a href="http://jisiksoft.com/publish.html">지식소프트 발간 서적</a></br>
        </br>
        <table>                                                    //---①
            <tr>                                                   //---②
                <th> How-to Series </th>                           //---③
                <th> 출간된 연도 </th>
                <th> 책 제목 </th>
            </tr>
            <tr>
                <td> 1 </td>                                       //---④
```

```
                <td> 2013 </td>
                <td> Big Number 연산 </td>
            </tr>
            <tr>
                <td> 2 </td>
                <td> 2014 </td>
                <td> 주식 분석 프로그램 만들기 </td>
            </tr>
            <tr>
                <td> 3 </td>
                <td> 2015 </td>
                <td> OCR 프로그래밍 </td>
            </tr>
        </table>
        </br>
        지식소프트는 매년 한권씩 출간하고 있습니다.</br>
        책 구매에 감사드립니다.</br>
    </body>
</html>
```

① <table></table> 태그를 사용하여 표가 만들어진다.

② <table></table> 태그 안에 4개의 <tr></tr> 태그가 있기 때문에 행Row이 4개 만들어진다.

③ <tr></tr> 태크 안에 3개의 <th></th> 태그가 있기 때문에 제목Head가 3개 만들어진다.

④ 하나의 행에서 3개의 <td></td> 태그가 있기 때문에 열Column이 3개 만들어진다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/7/table.html

그림 1-18 행과 열로 구분되어 만들어진 테이블 결과

[그림 1-18]에서 만들어진 표는 행과 열의 구분은 있지만 보기가 좋지 않다. 그래서 공백Space을 만들어주는 ' '를 추가하여 보기 좋게 만들어주는 코드가 [코드 1-2]이다. 결과는 [그림 1-19]와 같다. 디자인 속성을 만들어주는 CSS를 사용하지 않고도 HTML 만으로도 표를 좀더 보기 좋게 만드는 것은 가능하다. 하지만, 불필요하게 공백을 만들어주기 때문에 좋은 방법이 아니다. 여기서는 HTML 언어에서 중요하게 사용하는 공백Space을 어떻게 만들 수 있는지를 보여준다.

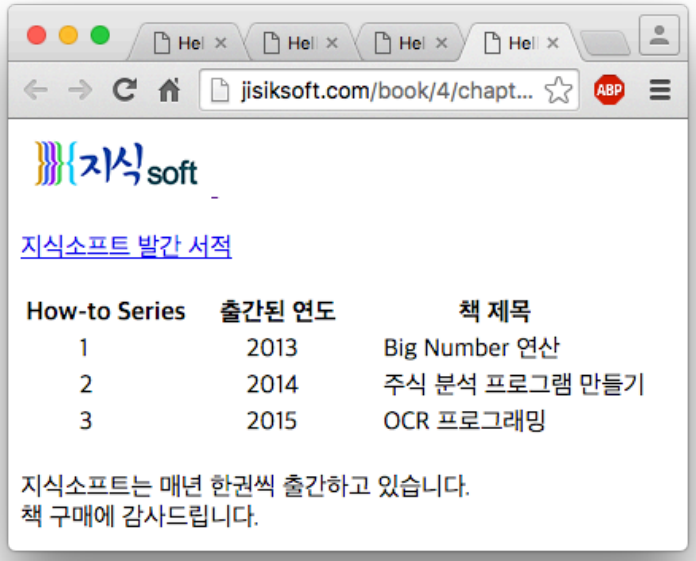
[코드 1-2] 공백을 만들어주는 특수문자 ' '를 사용한 Table 만들기 (/8/table.html)

```
<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8"></meta>
        <title>Hello JISIKSOFT</title>
    </head>
    <body>
        <a href="http://jisiksoft.com">
            <img src="./logo.png"></img>
        </a></br>
        </br>
        <a href="http://jisiksoft.com/publish.html">지식소프트 발간 서적</a></br>
        </br>

        <table>
            <tr>
                <th> How-to Series </th>
                <th>     출간된 연도 </th>
                <th> 책 제목 </th>
            </tr>
            <tr>
                <td>         1 </td>
                <td>         2013 </td>
                <td>       Big Number 연산 </td>
            </tr>
            <tr>
                <td>         2 </td>
                <td>         2014 </td>
                <td>      주식분석 프로그램만들기</td>
            </tr>
            <tr>
                <td>         3 </td>
                <td>         2015 </td>
                <td>       OCR 프로그래밍 </td>
            </tr>
        </table>
        </br>
```

```
        지식소프트는 매년 한권씩 출간하고 있습니다.</br>
        책 구매에 감사드립니다.</br>
    </body>
</html>
```

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/8/table.html

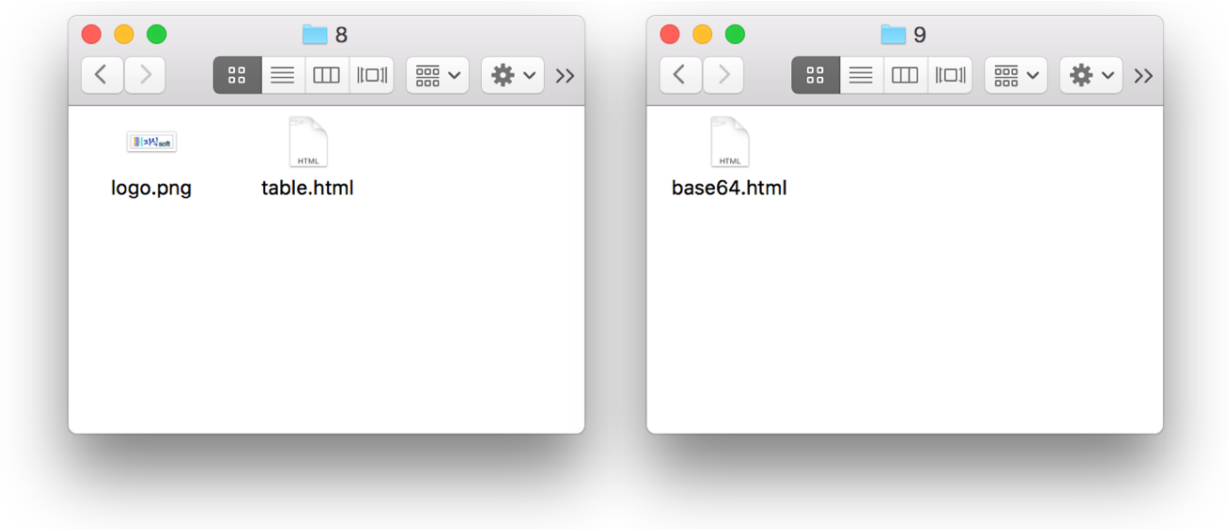
그림 1-19 공백을 만들어주는 특수문자 ' '를 사용하여 정렬한 결과



이미지들을 보여주기 위해서는 <img></img> 태그를 사용해서 이미지 파일을 연결하면 된다. Base64로 인코딩을 하면 이미지를 HTML파일 안에 넣는 것도 가능하다. Base64는 모든 데이터 파일을 문자 코드에 영향을 받지 않는 데이터로 변환해 주는 것인데, Base64로 인코딩이 된다는 것은 파일의 내용이 알파벳 52개(소문자 + 대문자)와 숫자 10개, 그리고 '+'와 '/' 2개를 포함한 총 64개의 육안으로 확인이 가능한 문자로 변환해 주는 것이며 크기를 맞추기 위해서 마지막에 '=' 문자가 들어가기도 한다. 브라우저에서는 이렇게 변환된 Base64 데이터를 약속된 규약Protocol에 맞추어서 화면에 보여준다.

[그림 1-20]은 왼쪽의 두 개의 파일(png와 html 파일)을 하나의 HTML 파일로 만들어 준 것이며, 결과는 [그림 21]처럼 같은 결과를 보여준다. 이미지를 html 파일 안에 넣는 것은 거의 사용하지 않는 방법이지만, 필요에 따라서 응용할 수 있기 때문에 알아두는 것이 좋다. 웹 프로그래머라면 Base64에 대한 기본 개념은 알고 있는 것이 좋다.

그림 1-20 같은 결과를 보여주지만 이미지 파일(PNG)이 HTML 코드 안으로 삽입될 수 있다.



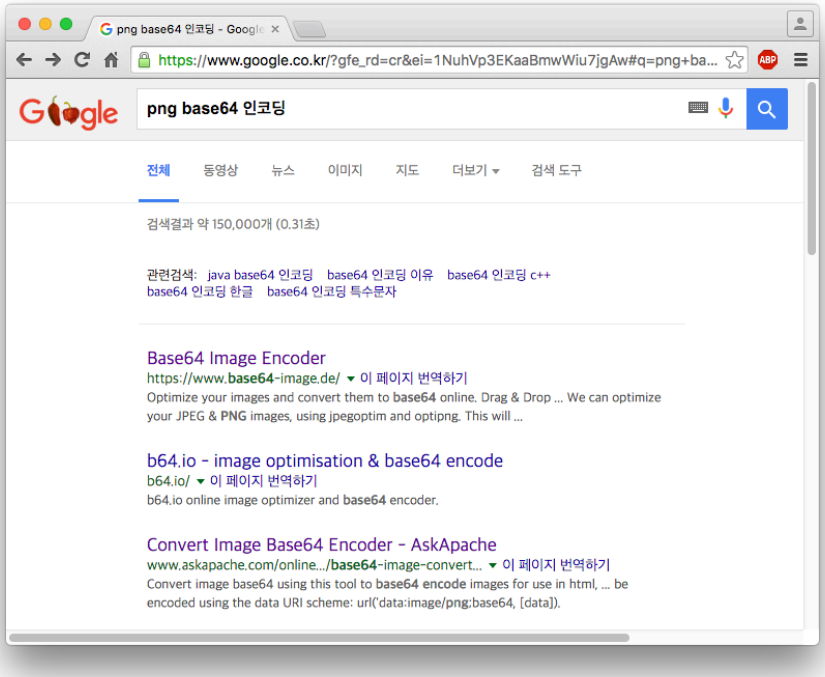
folder_8 확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/8/table.html

folder_9 확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/9/base64.html

그림 1-21 이미지를 Base64 로 변환하여 HTML 코드에 넣어도 이미지가 정상적으로 출력된다.

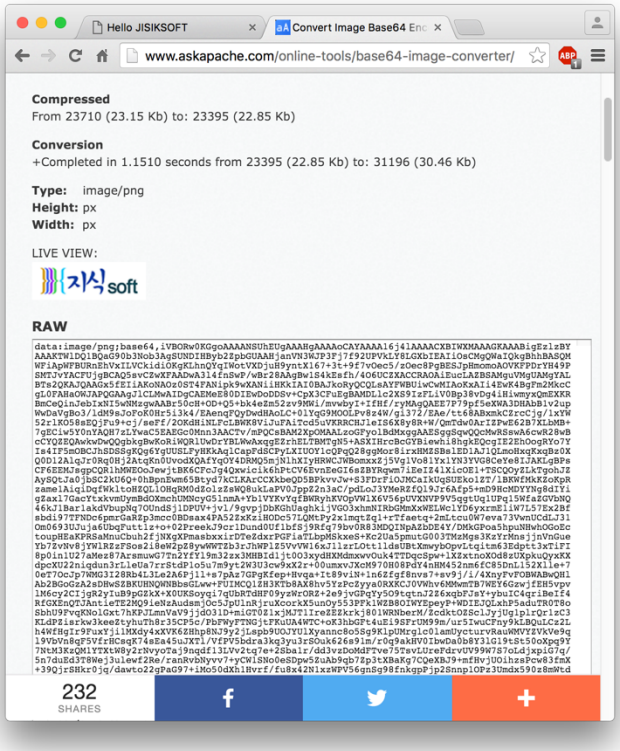
Base64로 인코딩을 제공하는 사이트들은 많다. Google에서 "png base64 인코딩"이라고 검색을 하면 다수의 사이트들이 있는데, 그 중에서 대표적인 사이트가 "AskApache"이다. 이 사이트에서 지식소프트 로고를 Base64로 변환한 결과를 [그림 1-23]에서 보여준다.

그림 1-22 Google 에서 "png base64 인코딩"을 검색한 결과



base64 인코딩 사이트 : http://www.askapache.com/online-tools/base64-image-converter/

그림 1-23 지식소프트 로고 이미지를 Base64 로 변환한 결과



[코드 1-3]은 Base64로 변환한 이미지를 <img></img> 태그의 src 속성값에 대입한 결과를 보여주는데, 데이터가 워낙 길기 때문에 중간 데이터는 생략했다.

[코드 1-3] 이미지 파일을 base64 로 인코딩해서 코드에 추가한다. (/9/base64.html)

```
<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8"></meta>
        <title>Hello JISIKSOFT</title>
    </head>
    <body>
        <a href="http://jisiksoft.com">
            <img src="
                oCAYAAAA16j4IAAAACXBIWXMAAAGKAAABigEzlzBYAAAKTWIDQ
                ##### 중간 데이터 생략 #####
                +RpVNS5JXH7AAAAABJRU5ErkJggg==">></img>
        </a></br>
        </br>
        <a href="http://jisiksoft.com/publish.html">지식소프트 발간 서적</a></br>
        </br>
        <table>
            <tr>
                <th> How-to Series </th>
                <th>     출간된 연도 </th>
                <th> 책 제목 </th>
            </tr>
            <tr>
                <td>         1 </td>
                <td>         2013 </td>
                <td>       Big Number 연산 </td>
            </tr>
            <tr>
                <td>         2 </td>
                <td>         2014 </td>
                <td>        주식 분석 프로그램 만들기
</td>
            </tr>
            <tr>
                <td>         3 </td>
                <td>         2015 </td>
                <td>       OCR 프로그래밍 </td>
            </tr>
        </table>
        </br>
        지식소프트는 매년 한권씩 출간하고 있습니다.</br>
        책 구매에 감사드립니다.</br>
    </body>
</html>
```

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.1/9/base64.html

HTML 언어만을 설명해도 책 한권의 분량이 된다. "How-to Series"는 모든 것을 설명하는 것이 아닌 이해를 위한 책이고, 프로그램을 만드는데 필요한 자료를 인터넷을 통하여 찾아서 조립한다는 개념이기 때문에 지금까지의 내용으로 HTML 언어를 이해하는 것이 좋다. HTML 언어를 공부하는 사람들 중에서는 HTML 언어에서 제공하는 수많은 태그들을 이해하려고 외우는 사람들도 있는데, 이 책에서 설명하는 몇 가지 태그를 충분히 이해하고 추가 태그들이 필요할 때 찾아서 사용하는 것이 더 좋은 방법이라고 필자는 생각한다. 한 가지 덧붙이자면, 기존에 많이 사용했던 태그들 중에는 표준에서 제외되는 태그들도 있기 때문에 HTML 언어의 모든 태그들을 알기 위해서 투자하는 시간에 JavaScript 언어를 좀더 연습하는 것이 웹 프로그래머에게 좋다고 생각한다. 처음 HTML 언어가 소개되었던 1990년대 중반은, 지금까지 설명된 HTML의 기본 개념으로 시작되었다. 그 당시에 웹 페이지는 텍스트 기반에 사진을 보여주는 정도의 인터넷 세상이었다. 웹 프로그래밍에서의 표준은 불필요한 것을 없애고 더 좋은 기능을 추가하면서 발전하고 있다. 예를 들면, 현재 많은 홈페이지들이 HTML에서 <iframe></iframe> 태그를 사용해서 홈페이지를 만들고 있는데, 최근에 발표된 HTML5에서는 <iframe></iframe> 태그를 제외했다. <iframe></iframe> 태그는 태그로 충분히 대체할 수 있는데, 이 책에서는 태그 위주로 대부분의 HTML 코드를 작성하기 때문에 태그의 중요성을 쉽게 이해할 것이다. 더하여 웹 프로그래밍을 좀더 쉽게 배우는 한가지 방법은 이 책의 부록에서 설명하는 브라우저의 "개발자 도구"를 통하여 모방하고 싶은 인터넷 사이트를 분석해서 필요한 부분을 가져다 사용하는 것이다. 실생활에서는 이 방법을 상당히 많이 활용하고 있다.

1.2 CSS

HTML 언어는 텍스트와 사진을 넣을 수 있지만 한계가 있다. 이것을 극복하기 위하여 HTML 언어의 기본인 태그에 디자인 속성을 넣는 것이 CSS인데, 태그에 직접 디자인 속성을 넣는 방법은 태그 속성 중의 하나인 'style'을 사용하는 것이다. 'style' 속성에는 위치, 크기, 색 등과 같은 많은 디자인 값을 부여할 수 있다. 즉, <body></body> 태그를 포함하여 그 안의 모든 태그들은 style로 디자인을 멋지게 말들 수 있다는 것이다. 필자는 사각형의 영역을 만들어 사용할 수 있는

 태그를 가장 좋아한다. 그 다음으로 많이 사용하는 것이 글자의 디자인을 위해서 사용하는 태그이다. 이전 장에서 우리는 다른 사이트로 이동하기 위해서 <a> 태그를 사용하였으나, 실제 태그로 <a> 태그 등과 같은 다양한 태그들의 동작을 수행할 수 있다. 이 방법은 다음 장에서 설명하는 JavaScript와 이벤트Event(ex: onclick)를 이해하면 된다. 이번 장에서는 우선 CSS의 기본 원리를 이해해야 하기 때문에 태그의 'style' 속성을 사용하여 디자인을 변경하는 방법부터 알아보기로 한다.

[코드 1-4]는 와 태그의 디자인을 변경하는 방법을 보여주는데, 태그에서는 폭(width)과 높이(height) 및 바탕색(background-color)으로 디자인을 만들고, 태그는 글자를 위한 디자인을 위하여 사용되기 때문에 글자 크기(font-size)와 글자 색(color)을 변경하였다. 화면에서 크기 값을 나타내는 픽셀Pixel(단위: px)과 색의 값을 정의하는 RGB(Red,Green,Blue)에 대해서는 "How-to Series"의 세 번째 책("OCR 프로그래밍" p23 Chapter_2)에 자세히 설명했으므로 여기서는 생략한다. [코드 1-4]의 결과는 [그림 1-24]에서 확인할 수 있는데, 첫 번째 는 가로 300px과 높이 100px의 값인 붉은색(Red: #ff0000) 사각형으로 만들었다. 이와 같은 원리로 두 번째 는 좀더 작은 녹색(Green: #00ff00) 사각형을 만들었다. 두 개의 태그들은 서로 다른 크기의 글자와 색을 갖는데, 여기서 중요한 것은 태그는 두 번째 태그가 옆이 아니라 아래에 만들어지는데 비해, 태그는 줄바꿈 없이 글자의 디자인만 변경한다. 프로그래머의 입장에서는 두 번째 태그가 아래에 새롭게 만들어지지 않는다면 텍스트를 위해서 사용하는 태그 대신에 를 사용하여도 무방할 것이다. 사용하는 태그들을 단순화 할수록 프로그래밍은 편하기 때문이다. 그러나, 텍스트의 디자인을 위해서는 항상 태그를 사용해야 한다고 이해하는 것이 필요하다.

[코드 1-4] style 속성을 적용하여 태그에 디자인 하기 (/1/square.html)

```
<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8"></meta>
        <title>Hello JISIKSOFT</title>
    </head>
```

```
    <body>
        <div style="width:300px;height:100px;background-color:#ff0000;"></div> //---①
        <div style="width:200px;height:80px;background-color:#00ff00;"></div>  //---②
        <span style="font-size:45px;color:#0000ff;">Hello </span>              //---③
        <span style="font-size:65px;color:#1646a7;">JisikSoft</span>           //---④
    </body>
</html>
```

① 사각형의 범위를 만드는 태그는 가로 300px, 세로 100px의 붉은색(#ff0000) 사각형을 만든다.

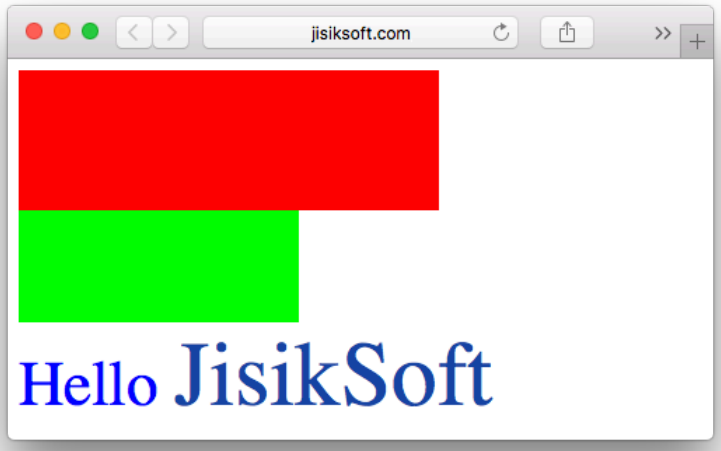
② 두 번째 사각형은 가로 200px, 세로 80px의 녹색(#00ff00) 사각형이 만들어진다.

③ 45px 크기의 파란색(#0000ff)의 'Hello' 글자가 출력된다.

④ 'Hello'보다 좀더 큰 65px 크기의 Navy(#1646a7) 'JisikSoft' 글자가 출력된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/1/square.html

그림 1-24 style 속성을 사용하여 HTML 태그에 디장인을 추가한 결과



HTML의 태그 안에 style 속성을 사용하여 디자인이 가능하다. 하지만, 페이지를 구성하는 모든 태그에 위와 같은 방법으로 디자인을 하게되면 코드를 읽기가 복잡해진다. (가독성이 안좋다.) 또한 중복되는 태그 속성도 많을 것이고, 다수의 프로그래머가 같이 일한다면 협업이 힘들어질 수 있다. 실제 업무에서는 코드만을 다루는 프로그래머들과 CSS 위주로 디자인을 하는 웹 디자이너들의 협업으로 이루어진다. 그래서 사용되는 방법이 HTML의 태그에 고유 이름을 부여하고 해당 이름에 대한 디자인을 CSS에 진행하게 된다. 태그들은 다수의 그룹으로 묶어질 수도 있고, 자신만

의 고유 식별자를 가질 수도 있다. [그림 1-25]에서는 태그를 구분짓기 위해서 사용하는 세 가지 속성인 class, id, name의 특징을 보여주는데, 실제 웹 프로그래밍에서 가장 많이 사용하는 것은 class와 id이다. class는 같은 성격의 태그들을 그룹으로 만드는 역할을 하며, id은 웹 프로그래밍 전체에서 오직 하나의 태그에서만 값을 갖게 된다. 클래스(class)와 id의 값을 정할 때는 하나의 규칙이 있는데, 첫 글자에는 숫자를 사용하면 안된다는 것이고, 이후에는 숫자와 '_' 기호 등을 사용하면 된다. 또하나의 규칙은 일반적인 프로그래밍에서 변수 이름을 정하듯이 해당 태그의 특징을 알 수 있는 값을 정하는 것이 좋다. CSS에서 사용할 때는 클래스(class) 값의 앞에 '.'(마침표)를 붙여주고 id 값의 앞에는 '#'(샵)을 붙여서 표현한다. '.'(마침표)와 '#'(샵)을 사용하는 규칙은 1.4장에서 설명하는 jQuery에서의 사용 방법과 동일하다. CSS가 정의된 이후에 만들어진 jQuery는 웹 프로그래밍에서의 규칙을 단순화하기 위해서 CSS와 같은 사용 규칙을 따른다.

그림 1-25 class 와 id 정의 예제와 CSS 에서 사용 방법

<class / id / name 예제>
class = 학과 / 대한민국 (큰 범위의 영역)
id = 학번 / 주민등록번호 (**하나만 존재**)
name = 성명 / 이름 (중복 가능)

<정의 예제>
class = math / korea (첫 글자는 알파벳 사용)
id = hak1004 / no1004 (첫 글자는 알파벳 사용)

<CSS 사용 방법>
class = .math / .korea (앞에 '.' 사용)
id = #hak1004 / #no1004 (앞에 '#' 사용)

이전 예제에서 설명한 와 태그들은 디자인에서 공통된 값을 갖는 것이 없기 때문에 id를 사용해서 모든 태그를 개별적으로 관리하는 것이 좋은데, [코드 1-5]는 <body></body> 안의 모든 태그에 id를 부여하고 style 속성값들을 <style></style> 태그를 사용하여 <head></head> 안으로 옮겼다. <style></style> 태그 안에 있는 내용들이 우리가 말하는 CSS라고 이해하면 된다. <body></body> 안의 태그들은 고유한 id 값으로 정의되었으며, CSS에서는 모든 id 값에 '#'(샵)을 사용해서 [코드 1-4]에서 사용된 디자인 값들을 동일하게 구현했다.

[코드 1-5]의 결과는 [그림 1-26]에서 보여지는 이전과 같은 결과를 보여준다.

[코드 1-5] CSS 기본 : style 을 <head></head> 태그 안으로 이동 (/2/squareCSS.html)

```
<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8"></meta>
        <title>Hello JISIKSOFT</title>

        <style>
            #square1 {                                          //---①
                width:300px;
                height:100px;
                background-color:#ff0000;
            }
            #square2 {
                width:200px;
                height:80px;
                background-color:#00ff00
            }
            #text1 {
                font-size:45px;
                color:#0000ff;
            }
            #text2 {
                font-size:65px;
                color:#1646a7;
            }
        </style>
    </head>

    <body>
        <div id="square1"></div>                                //---②
        <div id="square2"></div>
        <span id="text1">Hello </span>
        <span id="text2">JisikSoft</span>
    </body>
</html>
```

① id에 '#' 문자를 붙여서 해당 이름이 id 임을 확인한 후, 해당 id의 디자인 속성을 변경한다.

② 모든 태그들은 고유 id 값을 가지며, 하나의 id는 코드 안에서 하나의 태그에만 부여된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/2/squareCSS.html

그림 1-26 태그 안의 모든 style 값을 CSS 를 사용하여 동일하게 구현한 결과

일반적인 인터넷 사이트들을 "페이지 소스보기"로 확인하면 화려한 디자인임에도 불구하고 코드가 생각보다 길지 않을 때가 많다. 많은 태그들을 위와 같이 디자인하게 되면 <head></head> 태그 안의 내용이 상당히 길어진다. 그래서, 일반적으로 <style></style> 태그 안에서 정의된 CSS 내용들을 확장자가 .css인 개별 파일로 옮기는데, [코드 1-6]은 [코드 1-5]에서 <style></style> 태그의 내용을 'squareCSS.css' 파일로 옮기고 링크Link 기능을 하는 <link></link> 태그로 해당 파일을 연결했다.

[코드 1-7]은 'squareCSS.css' 파일의 내용을 보여주는데, <style></style> 태그 안의 내용이 그대로 옮겨졌음을 알 수 있다. 이와 같이, CSS 내용을 한 파일에만 넣어서 관리하는 것이 웹 프로그래밍에서 일반적으로 사용되는 방법이며, 이후에 이 책에서 구현되는 모든 코드의 CSS는 이와 같이 하나의 파일에 넣어서 관리된다.

[그림 1-27]은 브라우저에서의 결과를 보여주는데, 이전의 결과와 같다.

[코드 1-6] CSS 기본 : style 을 css 파일로 이동후 link 를 사용하여 연결 (/3/squareCSS.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./squareCSS.css"></link>                    //---①
    </head>
    <body>
        <div id="square1"></div>
        <div id="square2"></div>
```

```
        <span id="text1">Hello </span>
        <span id="text2">JisikSoft</span>

    </body>
</html>
```

① <link></link> 태그를 사용해서 .css 파일을 연결하는데, 연결되는 파일과의 관계 Relationship는 stylesheet 임을 나타낸다. CSS 파일을 이렇게 연결하는 것은 정해진 것이기 때문에 실제 코드 구현에서는 이전에 사용한 <link></link> 태그를 복사하고 href로 연결된 파일의 경로와 이름만 변경해주면 된다. 프로그래밍은 외우는 것이 아니라, 이해하고 응용하는 것임을 명심하자.

[코드 1-7] CSS 기본 : style 을 css 파일로 이동한 결과 (/3/squareCSS.css)

```
#square1 {
    width:300px;
    height:100px;
    background-color:#ff0000;
}
#square2 {
    width:200px;
    height:80px;
    background-color:#00ff00
}
#text1 {
    font-size:45px;
    color:#0000ff;
}
#text2 {
    font-size:65px;
    color:#1646a7;
}
```

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/3/squareCSS.html

그림 1-27 HTML 안에서 사용한 CSS 를 .css 확장자를 가진 파일로 이동해도 결과는 같다.

필자는 가능하다면 거의 모든 웹 프로그래밍을 만을 사용해서 단순하게 만들려고 하는 편이다. 가령 <img> 태그를 사용해서 사진을 불러오더라도 <img> 태그를 감싸고 있는 것은 태그를 사용하여 위치와 크기를 조절하는 편이다. 브라우저에서 를 사용하여 사각형 안에서 디자인을 하려고 하게 되면 많은 것을 기억할 필요 없이 다양한 표현이 가능하기 때문이다. 태그 안에는 또다른 태그를 삽입해서 디자인이 가능하기 때문에 지금부터 설명하는 CSS의 예제는 태그를 중심으로 위치를 이해하는 코드들을 구현할 것이다. 현재 이슈가 되고 있는 HTML5 언어에서는 이전 버전에서 많이 사용된 <frame></frame> 태그를 사용하지 않는데, 태그만으로 <frame></frame> 태그의 표현이 가능하기 때문이다.

[코드 1-8]은 4개의 태그들을 만들어서 CSS 속성으로 위치와 크기를 정의하는 예제이며 모든 태그들은 공통적으로 'square'라는 클래스명과 개별적으로 id를 하나씩 가지고 있다. HTML 파일에서 'position.css'라는 CSS 파일을 링크하는데 [코드 1-9]는 CSS 파일의 내용을 보여준다. 'square'는 클래스명이기 때문에 앞에 '.'(마침표) 기호를 붙였고, 나머지 4개의 id에는 '#'(샵) 기호를 사용한다. 'square' 클래스에는 공통적으로 사각형 크기와 글자 크기, 그리고 글자 색이 정의되었으며, 개별 id에는 사각형 색이 다르게 설정되었다.
[그림 1-28]에서는 위치가 정해지지 않은 네 개의 사각형이 차례대로 정렬된 결과를 볼 수 있다. 즉, 태그들의 기본 속성은 아래로 정렬되는 특징이 있다.

[코드 1-8] 사각형의 위치를 확인하기 위한 코드 (/4/position.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./position.css"></link>
    </head>
    <body>
        <div class="square" id="s1"> 1 </div>
        <div class="square" id="s2"> 2 </div>
        <div class="square" id="s3"> 3 </div>
        <div class="square" id="s4"> 4 </div> </body>
</html>
```

[코드 1-9] 위치를 지정하지 않았을 경우 (/4/position.css)

```
.square {
	width:100px;
	height:70px;
	font-size:30px;
	color:#ffff00;
}
#s1 {
	background-color:#1646a7;
}
#s2 {
	background-color:#3868c9;
}
#s3 {
	background-color:#5a8aeb;
}
#s4 {
	background-color:#7cacfd;
}
```

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/4/position.html

그림 1-28 위치가 정해지지 않은 태그들은 아래로 순서대로 배치된다.

이제부터는 HTML 파일에 변화를 주지 않고, CSS 정보만을 변경해서 로 만들어진 사각형의 위치를 이해하려고 한다. [코드 1-10]에서는 모든 4개의 태그에 float 속성값을 left로 설정하는데, [그림 1-29]와 같은 결과를 볼 수 있다. 즉, 4개의 로 만들어진 사각형들은 차례대로 왼쪽으로 정렬

된다. 이와 같은 방법으로 이전에 아래로만 정렬되던 사각형들을 가로로 정렬하는 것이 가능해진다.

[코드 1-10] float 의 left 를 사용하여 왼쪽부터 정렬 (/5/floatLeft.css)

```
.square {
      width:100px;
      height:70px;
      font-size:30px;
      color:#ffff00;
      float:left;                                                    //---①
}
#s1 {
      background-color:#1646a7;
}
#s2 {
      background-color:#3868c9;
}
#s3 {
      background-color:#5a8aeb;
}
#s4 {
      background-color:#7cacfd;
}
```

① 모든 태그들이 공통적으로 그룹지어진 'square' 클래스 속성에 왼쪽으로 정렬시키는 'float:left'를 적용하였다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/5/position.html

그림 1-29 'float:left' 설정으로 가로로 차례대로 사각형이 정렬된다.

'float' 속성값에 left 값이 있다면, 당연히 right 값도 존재할 것이며, [코드 1-11]과

같이 'float:right;'으로 설정하면 [그림 1-30]과 같이 오른쪽부터 차례대로 정렬된 결과를 볼 수 있다.

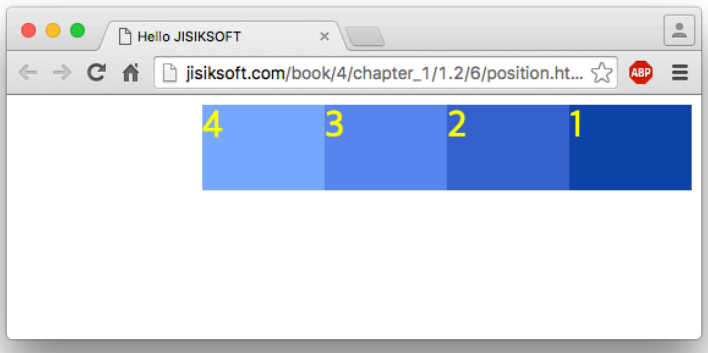
[코드 1-11] float 의 right 을 사용하여 오른쪽부터 정렬 (/6/floatRight.css)

```
.square {
      width:100px;
      height:70px;
      font-size:30px;
      color:#ffff00;
      float:right;                                                    //---①
#s1 {
      background-color:#1646a7;
}
#s2 {
      background-color:#3868c9;
}
#s3 {
      background-color:#5a8aeb;
}
#s4 {
      background-color:#7cacfd;
}
```

① 모든 의 float을 right으로 설정하면 오른쪽으로 차례대로 정렬된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/6/position.html

그림 1-30 'float:right' 설정으로 오른쪽부터 차례대로 사각형이 정렬된다.



'float'으로 설정된 에서 해당 설정값을 없애는 방법은 'clear' 속성을 사용하는 것이다. 만약 'float:left'로 설정되었다면 'clear:left'로 해제하고, 'float:right'으로 설정되었다면 'clear:right'으로 'float'의 설정을 해제하면 된다. [코드 1-12]는 모든 태그들이 'float:left'로 설정되었는데, 세 번째

에서만 'float'의 설정값을 'clear:left'를 사용해서 해제했다.

[그림 1-31]은 결과를 보여주는데, 첫 번째와 두 번째는 왼쪽으로 정렬되었고, 세 번재는 'float:left'가 적용되지 않은 것으로 인식되어 아래로 내려갔다. 이후에 따라오는 네 번째 사각형은 'float:left'의 영향으로 다시 세 번째 사각형의 우측에 정렬되었다.

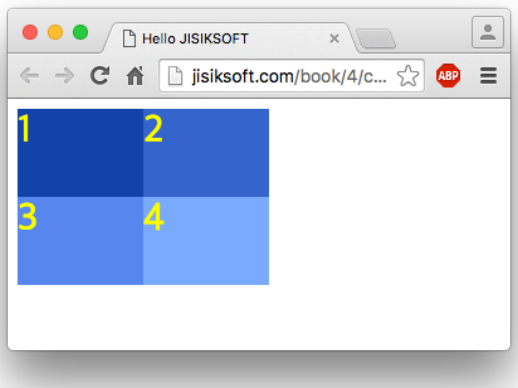
[코드 1-12] float 의 right 을 사용하여 오른쪽부터 정렬 (/7/clearLeft.css)

```
.square {
      width:100px;
      height:70px;
      font-size:30px;
      color:#ffff00;
      float:left;
}
#s1 {
      background-color:#1646a7;
}
#s2 {
      background-color:#3868c9;
}
#s3 {
      clear:left;                                                    //---①
      background-color:#5a8aeb;
}
#s4 {
      background-color:#7cacfd;
}
```

① 'square' 클래스에서 공통적으로 적용된 'float:left' 속성을 's3' id 값을 가진 세 번째 에서는 'clear:left'를 사용하여 설정을 해제했다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/7/position.html

그림 1-31 'float:left' 설정이 해제된 세 번째 사각형이 아래로 내려갔다.



태그들의 위치는 브라우저에서 프로그래머 마음대로 정할 수 있다. 브라우저의 가로축을 X-축으로 인식하고 세로축을 Y-축으로 생각하면 되는데, 모든 UI 프로그래밍의 개념은 오른쪽으로 갈수록 X 값이 증가하고 아래로 내려갈수록 Y 값이 증가하는 개념이다. CSS에서는 오른쪽으로 증가하는 위치를 왼쪽(left)에서 몇 픽셀Pixel 떨어져 있는지로 표현하며, 아래쪽으로 내려가는 위치를 위(top)에서 몇 픽셀Pixel 차이가 나는지로 표현한다. 예를 들면, 'left:300px'이라고 하면 오른쪽으로 300 픽셀Pixel 이동한 것이고, 'top:200px'이라고 하면 아래로 200 픽셀Pixel 내려간 것이다.

위치에서 가장 중요한 것이 위치(position) 속성이며, position 속성이 어떻게 정의되었는지에 따라서 태그의 위치를 프로그래머가 원하는대로 옮길 수 있다. 먼저 다루는 것이 절대값(absolute)으로 설정해서 브라우저의 왼쪽 위를 (0,0)으로 기준으로 잡고 top과 left를 사용하여 이동시키는 방법이다. 태그를 주로 사용하여 웹 응용프로그램을 만드는 프로그래머가 많이 사용하는 것이 'position:absolute' 설정값인데, 특정 위치에서 보여지는 내용들을 디자인하기에 아주 좋다.
[코드 1-13]은 모든 사각형들을 'position:absolute'로 정의하고 각각의 사각형은 'top'과 'left'를 사용해서 특정 위치로 옮겨지도록 CSS를 설정하였다.
[그림 1-32]는 모든 사각형들이 CSS에서 설정된 위치로 옮겨진 결과를 보여준다.

[코드 1-13] 위치(position)를 절대값(absolute)로 지정했을 경우 (/8/position.css)

```
.square {
    position:absolute;                                    //---①
    width:100px;
    height:70px;
    font-size:30px;
    color:#ffff00;
}
#s1 {
    top:150px;                                            //---②
    background-color:#1646a7;
}
#s2 {
    left:200px;                                           //---③
    background-color:#3868c9;
}
#s3 {
    top:200px;
```

```
    left:300px;
    background-color:#5a8aeb;
}
#s4 {
    background-color:#7cacfd;
}
```

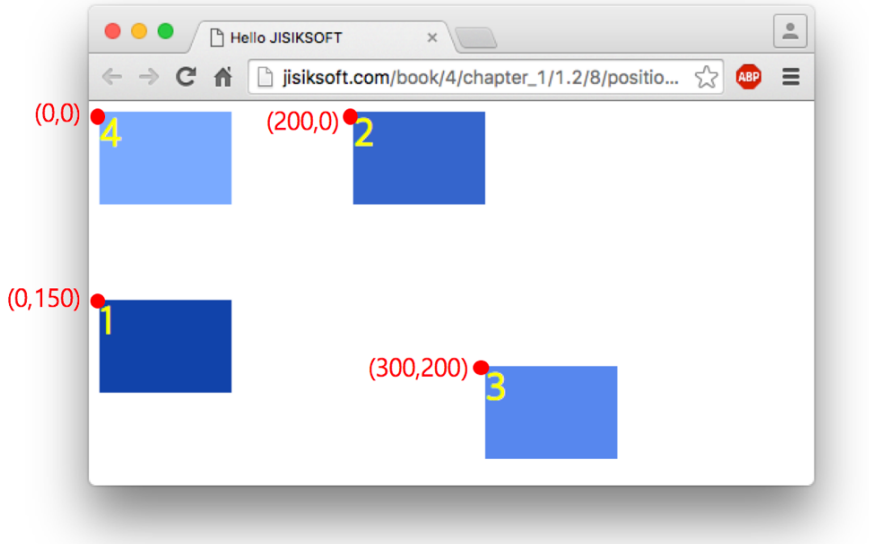
① 모든 사각형을 'position:absolute'로 설정해서 절대값만큼 위치를 이동시키게 된다.

② 첫 번째 사각형은 브라우저에서 위쪽(top) 끝에서 아래로 150 픽셀Pixel 만큼 이동한다.

③ 두 번째 사각형은 브라우저에서 왼쪽(left) 끝에서 오른쪽으로 200 픽셀Pixel 만큼 이동한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/8/position.html

그림 1-32 'position:absolute'를 사용하여 원점(0,0) 위치에서 절대값 만큼 사각형이 이동한 결과



위치(position) 속성에 절대값(absolute)와 함께 자주 사용되는 속성은 상대값(relative)이다. [그림 1-33]을 보면 쉽게 이해되는데, 'position' 속성이 정의되지 않으면 그림의 왼쪽과 같이 배열된다. 세로로 나란히 정렬된 4개의 사각형을 'position:relative'로 설정하고 'top'과 'left'의 속성으로 위치를 변경하게 되면 세로로 나란히 정렬된 곳에서 시작하여 위치가 변경된다.

[코드 1-14]에서 설정된 값만큼 이동된 결과를 [그림 1-33]에서 보여준다. 실제 웹 프로그래밍에서 'position:relative'는 상당히 많이 사용되는데, 이 책에서의 예제는 이해를 쉽게 하기 위해서 세 번째 사각형을 '-100'만큼 이동했지만, 웹 프로그래밍

에서 음수만큼 위치를 이동하는 일은 거의 없다.

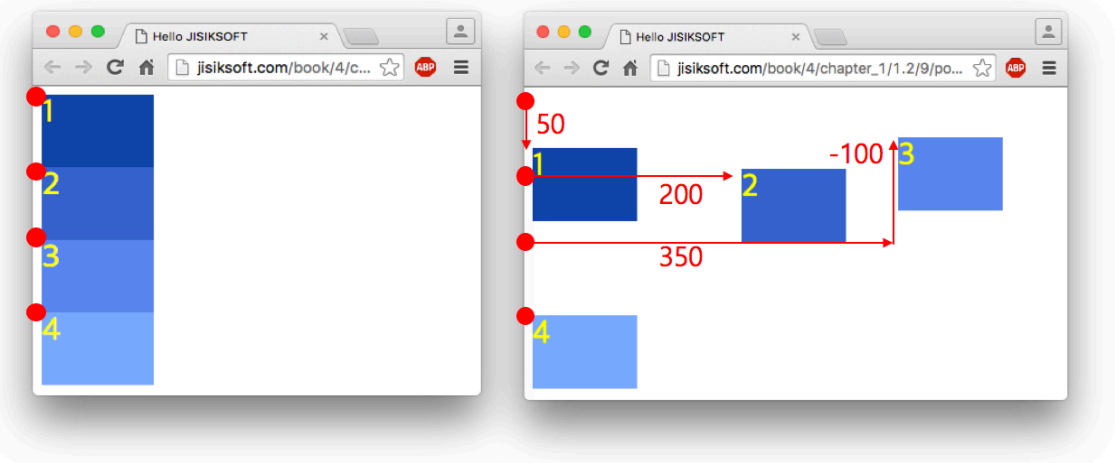
[코드 1-14] 위치(position)를 상대값(relative)로 지정했을 경우 (/9/position.css)

```
.square {
      position:relative;                                          //---①
      width:100px;
      height:70px;
      font-size:30px;
      color:#ffff00;
}
#s1 {
      top:50px;
      background-color:#1646a7;
}
#s2 {
      left:200px;
      background-color:#3868c9;
}
#s3 {
      top:-100px;
      left:350px;
      background-color:#5a8aeb;
}
#s4 {
      background-color:#7cacfd;
}
```

① 모든 사각형의 위치(position) 속성을 상대값(relative)로 설정했다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/9/position.html

그림 1-33 'position:relative'로 차례대로 배열된 위치에서 상대값 만큼 이동한 결과



웹 사이트들을 검색하다보면 브라우저의 스크롤을 사용하여 페이지를 위/아래로 움직여도 항상 고정되어서 움직이지 않는 정보들을 발견할 때가 있다. 자주 쓰이지는

않는 기술이지만 항상 화면에 노출해야 하는 경우에는 현재 보고 있는 브라우저 창의 특정 위치에 고정하게 되는데, 이때 사용하는 위치(position) 속성값은 고정값(fixed)이며, 'position:fixed'로 설정한다.

[코드 1-15]는 사각형들을 브라우저의 특정 위치에 고정시키는 코드이며, [그림 1-34]를 보게되면 오른쪽의 스크롤을 아래로 내려도 사각형의 위치들은 변하지 않는다.

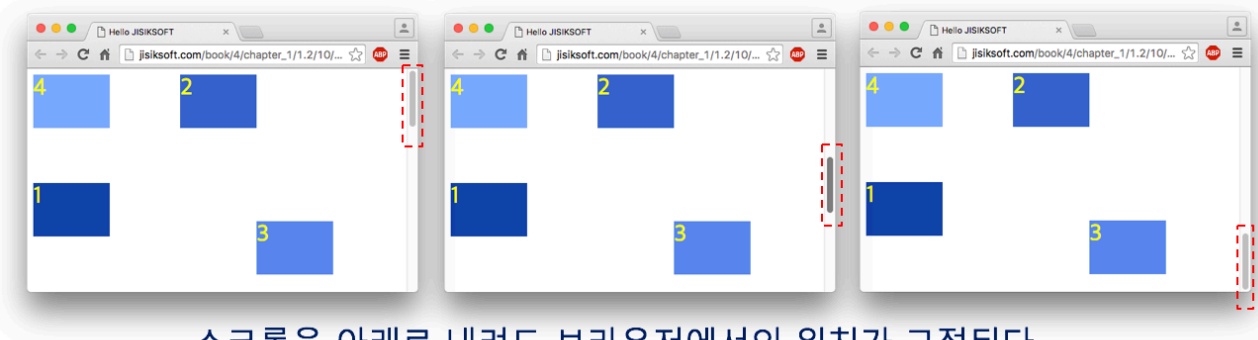
[코드 1-15] 위치(position)를 고정값(fixed)로 지정했을 경우 (/10/position.css)

```
.square {
    position:fixed;                                    //---①
    width:100px;
    height:70px;
    font-size:30px;
    color:#ffff00;
}
#s1 {
    top:150px;
    background-color:#1646a7;
}
#s2 {
    left:200px;
    background-color:#3868c9;
}
#s3 {
    top:200px;
    left:300px;
    background-color:#5a8aeb;
}
#s4 {
    background-color:#7cacfd;
}
```

① 모든 사각형의 위치(position) 속성을 고정값(fixed)으로 설정했다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/10/position.html

그림 1-34 'position:fixed'는 브라우저를 이동해도 항상 보여지는 화면에서 사각형들이 고정된다.



스크롤을 아래로 내려도 브라우저에서의 위치가 고정된다.

 태그는 사각형을 만드는데, 만들어진 사각형 안에 또다른 사각형을 넣을 수 있다. 이런 개념으로 태그 안에 다수의 태그를 넣게 되면 처음에 만들어진 의 위치를 변경하는 것만으로도 그 안에 있는 들의 위치는 종속되게 된다. 프로그래밍을 하다보면 부모(Parent)와 자식(Child)의 개념이 종종 나오곤 하는데, HTML 언어에서도 전체를 감싸고 있는 태그는 부모Parent라고 할 수 있고, 그 안에 종속되어지는 것을 자식Child 태그로 본다.

[코드 1-16]은 다수의 태그들을 감싸고 있는 'parent'라는 id를 가진 태그를 만들었으며, 이 'parent'의 위치를 조정하는 것만으로도 안에 있는 태그들의 위치가 변경되는 것을 알수 있다.

[코드 1-17]에서 'parent' id의 CSS 속성에서 위치를 변경하였으며 [그림 1-35]에서 결과를 보여주는데, 회색으로 사각형의 위치가 변경되면서 그 안의 사각형들의 위치도 동일하게 이동되었다. [코드 1-17]의 코드는 [코드 1-13]에 '#parent {}'의 속성이 추가되었다.

[코드 1-16] div 안에 포함된 하위 div 는 위치가 종속된다. (/11/position.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./position.css"></link>
    </head>

    <body>
        <div id="parent">                                          //---①
            <div class="square" id="s1"> 1 </div>
            <div class="square" id="s2"> 2 </div>
            <div class="square" id="s3"> 3 </div>
            <div class="square" id="s4"> 4 </div>
        </div>
    </body>
</html>
```

① 4개의 태그들을 감사는 'parent' id를 가진 상위 태그가 추가되었다.

[코드 1-17] 상위(부모 객체) div 가 추가된 css 설정 (/11/position.css)

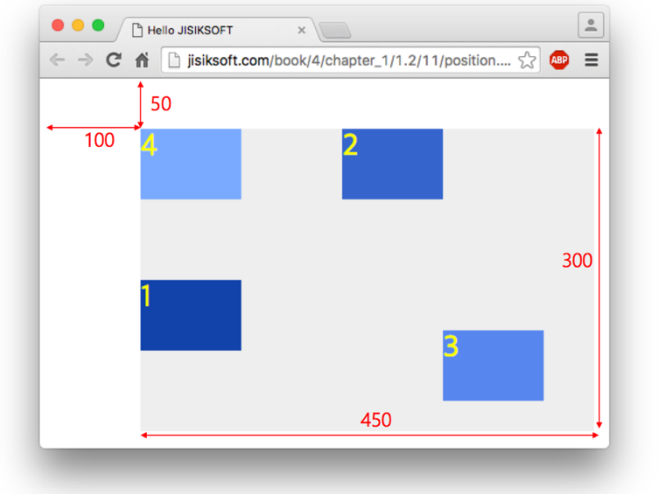
```
#parent {
```

```
        position:absolute;                                          //---①
        top:50px;
        left:100px;
        width:450px;
        height:300px;
        background-color:#eeeeee;
}
.square {
        position:absolute;
        width:100px;
        height:70px;
        font-size:30px;
        color:#ffff00;
}
#s1 {
        top:150px;
        background-color:#1646a7;
}
#s2 {
        left:200px;
        background-color:#3868c9;
}
#s3 {
        top:200px;
        left:300px;
        background-color:#5a8aeb;
}
#s4 {
        background-color:#7cacfd;
}
```

① 전체를 감싸고 있는 'parent' id를 가진 태그의 위치(position)를 절대값(absolute)으로 아래로 50 픽셀Pixel, 오른쪽으로 100 픽셀Pixel만큼 이동시키다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/11/position.html

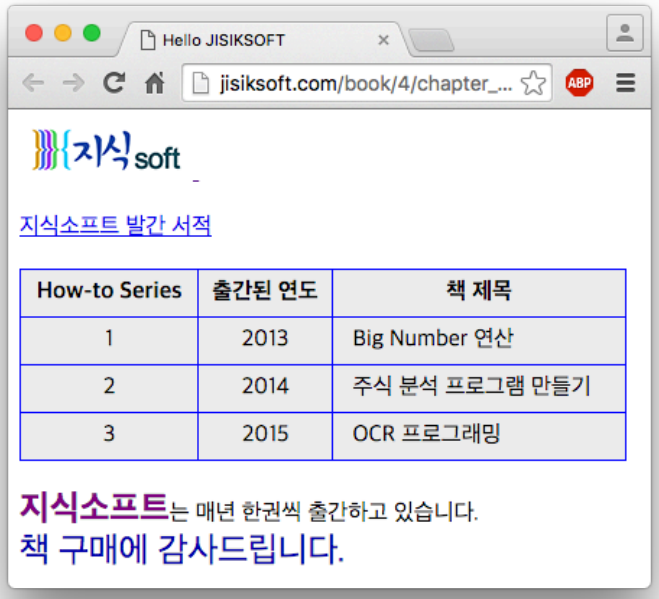
그림 1-35 회색 사각형이 이동하면 그 안에 있는 모든 사각형이 같이 이동한다.



대부분의 웹 페이지들은 멈춰있는 페이지이지만, 간혹 브라우저에서 움직이는 화면들을 접하게 된다. 동적인 화면을 만드는데 위치가 만약 움직인다면 태그로 감싸서 위치를 변경하는 웹 프로그램일 가능성이 크다. 상당히 방대한 내용의 CSS에 대해 필자는 태그의 중요성을 강조하면서 위치(position)에 중심을 두었는데, 프로그래머의 관점에서 보면 웹 프로그래밍의 기술 발전을 위해서는 를 사용하여 코드를 자유롭게 구현하는 것이 중요하기 때문이다. 'Chapter 2' 이후부터 이 책에서 구현하는 웹 프로그래밍은 중심으로 화면의 구조를 생각하는 것이 좋다. 브라우저에서 사진이 아닌 CSS의 멋진 디자인 처리를 구현하고 싶다면 이 책의 부록에서 소개하는 '개발자 도구'(ex: firebug)를 사용하여 해당 CSS의 속성을 확인하면 된다.

이전 장 "1.1 HTML"에서 우리는 브라우저에서 표Table를 만드는 법을 <table></table> 태그를 사용하여 구현하였다. 이번에는 이전 장에서 구현했던 코드에 CSS 디자인을 추가해서 [그림 1-36]에서와 같은 결과를 얻고자 한다. 일반적인 태그에 CSS 디자인을 설정하는 방법과 클래스Class 값으로 다수의 이름을 부여하는 방법이다.

그림 1-36 HTML 표(table)와 텍스트에 CSS 디자인을 적용한 결과 미리 보기



[그림 1-37]은 태그에도 CSS를 적용할 수 있는 예제를 보여준다. 클래스Class는 '.'(마침표)를 사용하고 id는 '#'(샵)을 이름앞에 사용하지만, 태그는 어떠한 문자도 앞에

붙이지 않고 태그 이름을 그대로 사용한다. 또한 ','(콤마)를 사용하여 다수의 태그들에 디자인을 일괄 적용할 수 있다. 즉, "table, th, td {}"와 같이 CSS에서 함께 정의함으로서 세 개의 태그들(<table></table>, <th></th>, <td></td>)을 동시에 디자인을 변경할 수 있다.

그림 1-37 CSS 에서 태그, 클래스, id 이름 사용 방법

```
<CSS : 태그Tag 설정 방법>
table {
    border-collapse: collapse;
}
table, th, td {
    height: 30px;
    text-align: center;
    border: 1px solid blue;
    background-color: #eeeeee;
}
```

[그림 1-38]은 CSS 파일에서 클래스Class 이름을 특징을 가지고 정의한 후, HTML 코드의 태그와 같은 곳에서 클래스 값을 중복사용하는 방법을 보여주는데, 많은 텍스트가 다양한 디자인 속성을 개별적으로 갖게 된다면 이와 같이 사용하는 것이 유용하다. 필자는 텍스트의 디자인을 CSS에서 미리 다양하게 설정한 이후에 태그의 클래스 이름으로 이와 같이 정의하여 사용하는 편인데 생각보다 편리한 점이 많다.

그림 1-38 다수의 클래스 값들을 중복 사용이 가능

```
1. CSS 클래스 설정
.purple { color: #800080; }
.bold { font-weight: bold; }
.big { font-size: 24px; }

2. 다수의 클래스값 정의
<span class="purple bold big">지식소프트</span>
```

[코드 1-18]은 1.1장에서 구현한 [코드 1-2]에 CSS 디자인을 추가하기 위하여 다수의 태그들에게 클래스Class와 id 값을 부여하였다. 또한 마지막 부분의

 태그에는 여러 개의 클래스 값을 함께 사용해서 id를 사용하지 않으면서도 개별적인 디자인의 적용을 가능하게 구현하였다.

[코드 1-19]에서는 CSS의 설정을 보여주는데, 태그에 개별적으로 디자인을 적용하였으며, 마지막에는 하나의 속성만을 가진 다수의 클래스Class들을 설정하였다. 이와 같이 하나의 속성만을 가진 클래스의 이름을 정의할 때는 이름만으로도 해당 클래스의 특징을 알수 있게 만들어 주어야 한다.

[코드 1-18] div 안에 포함된 하위 div 는 위치가 종속된다. (/12/tableCSS.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./tableCSS.css"></link>
    </head>
    <body>
        <a href="http://jisiksoft.com">
            <img src="./logo.png"></img>
        </a></br>
        </br>
     <a class=”pointer” href="http://jisiksoft.com/publish.html">지식소프트 발간서적</a></br>
        </br>
        <table>
            <tr>
                <th id="th_1"> How-to Series </th>                    //---①
                <th id="th_2"> 출간된 연도 </th>
                <th id="th_3"> 책 제목 </th>
            </tr>
            <tr>
                <td> 1 </td>
                <td> 2013 </td>
                <td class="td_3">    Big Number 연산 </td>          //---②
            </tr>
            <tr>
                <td> 2 </td>
                <td> 2014 </td>
                <td class="td_3">    주식 분석 프로그램 만들기 </td>
            </tr>
            <tr>
                <td> 3 </td>
                <td> 2015 </td>
                <td class="td_3">    OCR 프로그래밍 </td>
```

```
            </tr>
        </table>
        </br>
        <span class="purple bold big">지식소프트</span>는 매년 한권씩
                                        출간하고 있습니다.</br>
        <span class="blue big">책 구매에 감사드립니다.</span></br>          //---③
    </body>
</html>
```

① 행의 폭을 특정 크기로 설정하기 위하여 개별 id로 'th_1', 'th_2', 'th_3' 세 개의 값을 부여했으며, <th></th> 태그에서만 폭을 정하여도 아래에 출력되는 나머지 <td></td> 태그들의 폭이 자동으로 조절된다.

② 모든 셀Cell의 텍스트 위치를 중앙Center으로 설정했으며, 세 번째 행을 나타내는 세 번째 <td></td> 태그들을 클래스Class를 사용하여 왼쪽으로 정렬되게 하기 위해서 'td_3' 클래스 값을 사용한다.

③ 태그 안의 글자들을 여러 개의 클래스 값들을 사용하여 쉽게 디자인을 변경한다. 즉, "purple bold big"은 글자를 보라색의 굵고 큰 텍스트로 만들어주고, "blue big"은 파란색의 큰 텍스트를 만든다.

[코드 1-19] div 안에 포함된 하위 div는 위치가 종속된다. (/12/tableCSS.css)

```
table {                                                          //---①
    border-collapse: collapse;
}
table, th, td {                                                  //---②
    height: 30px;
    text-align: center;
    border: 1px solid blue;
    background-color: #eeeeee;
}
#th_1 {                                                          //---③
    width:120px;
}
#th_2 {
    width:90px;
}
#th_3 {
    width:200px;
}
.td_3 {
    text-align: left;                                            //---④
}
```

```
.gray { color: #999999; }                                    //---⑤
.red { color: #dd0000; }
.green { color: #00aa00; }
.blue { color: #0000aa; }
.pink { color: #ff0080; }
.ocher { color: #dd7700; }
.orange { color: #ffa500; }
.purple { color: #800080; }
.under { text-decoration: underline; }                       //---⑥
.arial { font-family: arial; }                               //---⑦
.bold { font-weight: bold; }                                 //---⑧
.big { font-size: 24px; }                                    //---⑨
.normal { font-size: 17px; }
.small { font-size: 14px; }
.line_20 { line-height:2.0em; }
.pointer { cursor: pointer; }                                //---⑩
```

① 표Table에 선을 그려주기 위해서 table 태그의 'border-collapse' 속성을 'collapse'로 설정했는데, 표Table와 표안에 있는 셀Cell들의 테두리 선들이 겹쳐서 그려지게 하는 역할을 한다.

② table, th, td 태그들의 CSS를 설정하는데 셀Cell의 높이는 30 픽셀Pixel이고, 텍스트는 중앙에 위치하고 가는 선을 그려주며, 바탕은 옅은 회색이다. (RGB에 관련한 내용은 "How-to Series" 세 번째 책 <OCR 프로그래밍> p23 참조)

③ 표Table의 첫 번째 행의 폭을 120 픽셀Pixel, 두 번째 행의 폭을 90 픽셀Pixel, 세 번째 행의 폭을 200 픽셀Pixel 로 설정한다.

④ 세 번째 행의 내용이 들어가는 세 번째 <td></td> 태그들의 텍스트들은 왼쪽으로 정렬된다.

⑤ 단일 속성을 갖는 클래스Class들을 정의하는데 색의 값들을 설정하고, 클래스명은 색을 알수 있는 색의 명칭을 사용하였다.

⑥ 텍스트에 밑줄을 그을 수 있는 CSS 설정을 'under' 클래스명으로 정했다.

⑦ 글자체를 arial로 설정하는 클래스를 정의했다.

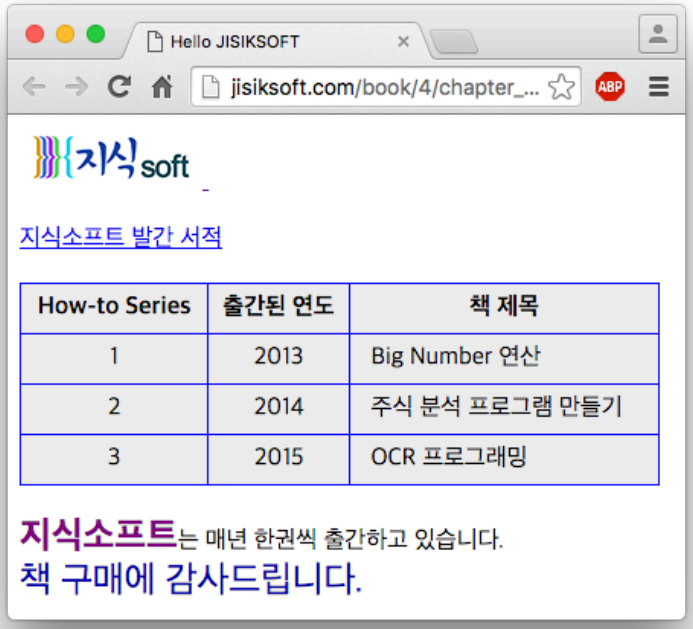
⑧ 글자를 두껍게 보여주는 bold 클래스를 정의하였다.

⑨ 글자의 크기는 특정 픽셀Pixel의 크기로 정한 수, 'big', 'normal', 'small'과 같이 세 가지 형태로 클래스명을 정했다.

⑩ 글자에 마우스를 갖다대면 클릭이 가능한 아이콘으로 변하게 하기 위해서 마우스 커서(Cursor)를 포인터(Pointer) 모양으로 설정하는 클래스다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.2/12/tableCSS.html

그림 1-39 HTML 표(table)와 텍스트에 CSS 디자인을 적용한 결과



CSS 내용만 설명해도 몇백 페이지 분량의 두꺼운 책을 만들수 있다. 그만큼 방대한 디자인 속성들이 있지만, "How-to Series"의 기본 개념은 필요할 때 찾아서 사용한다는 것이다. 다행히도 웹 프로그래밍은 카피Copy가 용이하다는 것인데, 어디선가 사용된 멋진 CSS 내용들을 그대로 갖다 써도 누가 뭐라 하지 않는다는 것이다. 일반적인 소프트웨어라면 소스 코드Source Code를 확인할 수 없기 때문에 핵심적인 내용을 가져올 수 없지만, 웹 프로그래밍에서는 모든 코드가 공개되어 있으니 누구든지 자신만의 소프트웨어를 만들 수 있는 좋은 환경이다.

1.3 JavaScript

HTML과 CSS를 이해하면 브라우저에서 보기 좋은 화면을 만들 수 있다. 그러나, 사용자 입장에서는 참여할 수 없는, 일방적인 화면에 불과하다. JavaScript는 아름답기만한 화면에 생명을 불어 넣는 역할을 하는 프로그래밍 언어라고 보면 된다. 간혹 Java를 붕어, JavaScript를 붕어빵이라고 비유하곤 한다. 이 둘은 이름만

비슷할 뿐 전혀 연관성이 없다. Java를 한다고해서 JavaScript를 할 수 없으며 단지 연관성을 찾는다면 변수의 선언이 쉽게 되어 있어서 C/C++ 보다 프로그래밍하기가 편한 정도이다. 예를 들면, C/C++에서는 실수형, 정수형, 문자형 등등의 변수를 꼭 구분해야 하는데, Java와 JavaScript는 이러한 규칙을 따르지 않고, 'var'Variable로만 변수를 선언하면 된다. Java는 일반적인 프로그래밍 언어처럼 컴파일Compile이라는 단계를 거쳐서 기계어로 변환된 실행파일을 만들지만, JavaScript는 기계어가 아니라 브라우저가 소스 코드를 해독해서 동작하는 스크립트Script언어이기 때문에 우리가 속도 차이를 크게는 느끼지 못하더라도 Java 보다는 느리게 실행된다. 간혹, JSP와 JavaScript의 차이를 이해하지 못하는 사람을 현장에서 볼 때가 있는데, JSP는 서버에서 동작하는 Java 언어를 사용하는 것이고, JavaScript는 단지 사용자 브라우저에서만 동작하는 언어임을 유념해야 한다.

이벤트 종류	설명
onclick	버튼을 마우스 왼쪽 버튼으로 누르면 다음 동작을 수행할 때 사용한다.
onmouseover	버튼 같은 곳에 마우스가 이동할 때 버튼의 색이 바뀌는 동작 등에서 사용한다.
onmouseout	버튼 밖으로 마우스가 이동할 때, 버튼의 색이 원래대로 바뀌는 등의 동작에서 사용한다.
onload	사이트에 처음 방문할 때 브라우저에서 모든 정보들을 받은 후에 자동으로 동작해야 하는 함수들은 onload 이벤트로 실행된다.
onresize	브라우저의 화면 창의 크기가 변경될 때 발생하며, 화면 크기가 변동하면 이미지의 크기가 이에 맞춰 변경되는 사이트에서 주로 사용한다.

사용자와 페이지는 버튼을 선택하는 등의 동작으로 교류를 한다. 여기서 이해해야 할 것이 이벤트Event인데, 마우스를 클릭하거나 마우스를 이동하는 동작들을 웹 페이지가 인식하고 그에 따른 처리를 하게 된다. 실제, 브라우저가 마우스나 키보드의 동작을 직접 인지할 수 있는 것이 아니라, 컴퓨터의 운영체제OS가 마우스의 위치를 정확히 알고 브라우저 위에 마우스가 있으면 그 정보를 해당 브라우저에 알려주는 것이다. 즉, 사용자의 행위를 브라우저는 운영체제를 통해서

받아들이는 것이며, 그에 따른 처리를 브라우저가 수행하는 것이니 브라우저도 하나의 어플리케이션인 셈이다. 예를 들면, 우리는 "How-to Series" 두 번째와 세 번째 책에서 Windows에서 동작하는 응용프로그램을 만드는 법을 배웠는데, Windows에서 동작하는 모든 브라우저들(Explorer, Chrome, Firefox 등)은 MFC로 만들어진 윈도우 응용프로그램이며 HTML, CSS, JavaScript 등의 스크립트Script 언어를 해석하는 기능을 한다. 검색 사이트에서 "HTML 이벤트 종류"라고 하면 많은 정보를 얻을 수 있다. 아래의 이벤트 정보는 자주 사용하는 몇 가지만을 정리한 것이다.

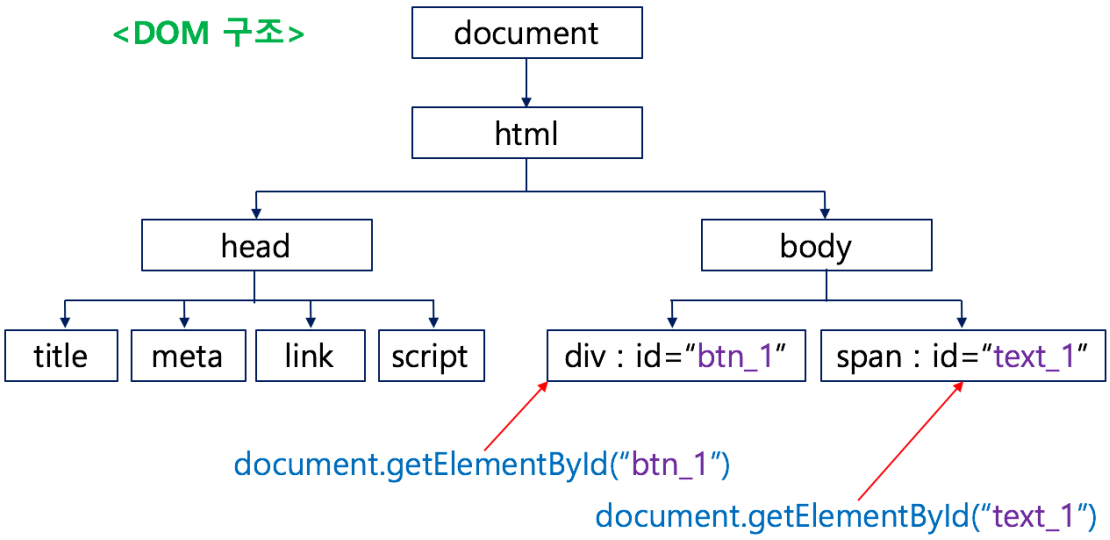
JavaScript는 변수Variable를 정의할 때는 앞에 'var'을 선언해주고, 함수Function를 정의할 때는 'function'을 사용한다.

[코드 1-20]은 와 태그을 사용하여 사각형과 텍스트를 마우스로 클릭하면 동작하는 JavaScript 함수를 구현한 것이며, <script></script> 태그 안에 JavaScript 언어를 사용하면 된다. HTML 언어의 모든 태그에는 'onclick'과 같은 이벤트들을 사용할 수 있으며, 가장 기본적인 방법을 이해하면 여러 가지로 응용하기가 쉬워진다. 웹 프로그래밍에서 가장 중요한 것은 DOM(Document Object Model) 구조를 이해하는 것인데 모든 태그는 시작과 끝이 있다는 것을 안다면 간단하다. 웹 프로그래밍은 HTML 언어에서 시작하는데, XML 언어의 한 종류인 HTML은 태그들로 이루어졌다.

[코드 1-20]을 DOM 구조로 그려보면 [그림 1-40]과 같은데 가장 상위에 있는 것이 'document'로서 HTML 파일 안에 있는 문서의 모든 것이다. 그 아래가 <html></html> 태그로서 모든 것을 감싸고 있다. 즉, 시작 태그와 종료 태그 안에 있는 것은 자신의 자식Child 태그들이다. 각각의 태그들로 감싸져 있는 것을 그 자신의 객체(Object)라고 보면 되는데, 각각의 태그들은 자신이 포함한 모든 것이 자신의 객체Object라고 해석하면 된다. 예를 들면, <head></head> 태그는 <title></title>, <meta></meta>, <link></link>, <script></script> 모두를 가지고 있는 객체Object다. JavaScript 언어에서 중요한 것이 DOM 구조에서 특정 객체Object를 필요에 따라 가져다가 사용할 수 있는데, 'getElementById()' 함수가 대표적이며 Object와 Element는 같은 의미로 해석하면 된다. 'getElementById()' 함수를 그대로 직역하면 "id로 객체Element를 가져온다.Get"로 풀이된다. [그림 1-40]에서 "document.getElementById("btn_1")"은 전체 문서인 document 객체에서 "btn_1"이라는 id를 가진 객체를 가져오고, "document.getElementById("text_1")는

"text_1"이라는 id를 가진 객체를 가져온다. 모든 태그는 id나 클래스Class 이름을 부여할 수 있는데, 이전 장에서는 id와 클래스를 사용하여 CSS 디자인에 사용했다면 JavaScript에서는 특정 객체에 직접 접근할 수 있는 방법으로 사용된다. DOM 구조를 이해한다는 것은 특정 객체Object로의 접근이 가능한 것을 이해한다는 것이고, 특정 객체에 접근할 수 있다는 것은 해당 객체의 내용이나 디자인 변경을 프로그래머가 원하는대로 변경할 수 있는 가능성을 주었다는 뜻으로 본다. 웹 프로그래밍이란 만들어진 내용을 보여주기만 하는 것이 아니라 사용자의 반응에 따라서 브라우저를 구성하고 있는 DOM의 객체Object에 변화를 주는 행위로 이해하는 것이 좋다. 화면의 특정 부분의 색을 변경하는 것 조차도 해당 객체에 접근해서 색을 변경해 주는 일련의 작업이 이루어진다. HTML 언어를 소개하는 책은 처음부터 DOM 구조를 설명하는데 DOM 구조가 어떻게 사용되는지를 이해하지 않고 이론을 먼저 접하게 되면 너무 추상적으로 느껴진다. 실제, CSS에서도 DOM 구조를 바탕으로 디자인 변경 작업이 이루어지는데, 웹 프로그래밍을 하는 프로그래머의 입장에서는 디자인 변경은 브라우저라는 프로그램에서 하는 일이라 CSS에서 직접적으로 해주는 일이 없기 때문에 JavaScript에서 DOM 구조를 이해하기 위해서 이번 장에서 설명했다.

그림 1-40 DOM(Document Object Model) 구조



[코드 1-20]은 onclick 이벤트에 의해서 JavaScript 함수가 실행되고 DOM 구조의 객체에 접근해서 사각형의 색을 변경하거나 경고 창을 발생시키는 것을 보여준다. 이 책에서는 JavaScript의 기본적인 언어의 특성을 모두 설명하지는 않지만, 필자의

이전 책들을 이해한 독자나 프로그래밍 언어를 조금이라도 접해본 독자라면 이해하는데 큰 어려움은 없을 것이다.

[코드 1-20] onclick 이벤트로 JavaScript 함수를 실행하는 코드 (/1/clickEvent.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./clickEvent.css"></link>

        <script>
            var arrayColor = [ '#770000', '#007700', '#000077' ];                    //---①
            var index = 1;                                                           //---②

            function runButton() {                                                   //---③
                index = (index + 1) % 3;
                document.getElementById("btn_1").style.backgroundColor = arrayColor[index];
            }

            function runText() {                                                     //---④
                alert("디버깅을 할때 가장 많이 사용하는 것이 alert() 함수입니다.");
            }
        </script>

    </head>

    <body>
        <div id="btn_1" onclick="runButton();"> I am a Button. Click Me!!! </div>          //---⑤
        <span id="text_1" onclick="runText();"> Click This To Get Message. </span>    //---⑥
    </body>
</html>
```

① JavaScript는 모든 변수는 'var'로 선언하는데, 배열Array조차도 변수이기 때문에 'var'만 사용하면 된다. Java와 같이 데이터형Data Type에 대해 항상 고려해야 하는 수고로움을 덜기 위해 JavaScript 언어를 설계한 의도가 엿보인다. JavaScript에서 배열은 '['와 ']' 문자들로 선언을 하며, 세 개의 RGB 값이 배열에 선언되었다.

② 위에서 정의된 배열 'arrayColor'의 순서Index를 정의하기 위하여 사용되는 변수로서, 브라우저에서 실행되면 처음에 사각형의 색이 '#007700'으로 설정되었기 때문에 배열의 두 번째(index = 1) 값으로 초기화했다. 'arrayColor' 배열은 세 개로 이루어

졌는데, 모든 프로그래밍 언어에서 배열의 순서Index는 0부터 시작한다.(arrayColor[0] = '#770000', ayColor[1] = '#007700', ayColor[2] = '#000077')

③ 사각형 버튼이 클릭되면 태그의 onclick 이벤트가 실행되며, onclick 이벤트는 runButton() 함수를 실행한다. 사각형의 색을 변경하고 배열의 다음 순서를 얻기 위해서 index 변수를 1 증가시키며, index 값이 3이 되면 0으로 되돌아 가기 위해서 나머지 연산자 '%'를 사용했다. 사각형의 객체를 가져오기 위해서 document.getElementById("btn_1)을 사용했다. 객체의 바탕색을 새로운 값으로 변경하기 위해서, 해당 객체의 스타일(style)의 바탕색(backgroundColor)을 새로운 색으로 변경한다.

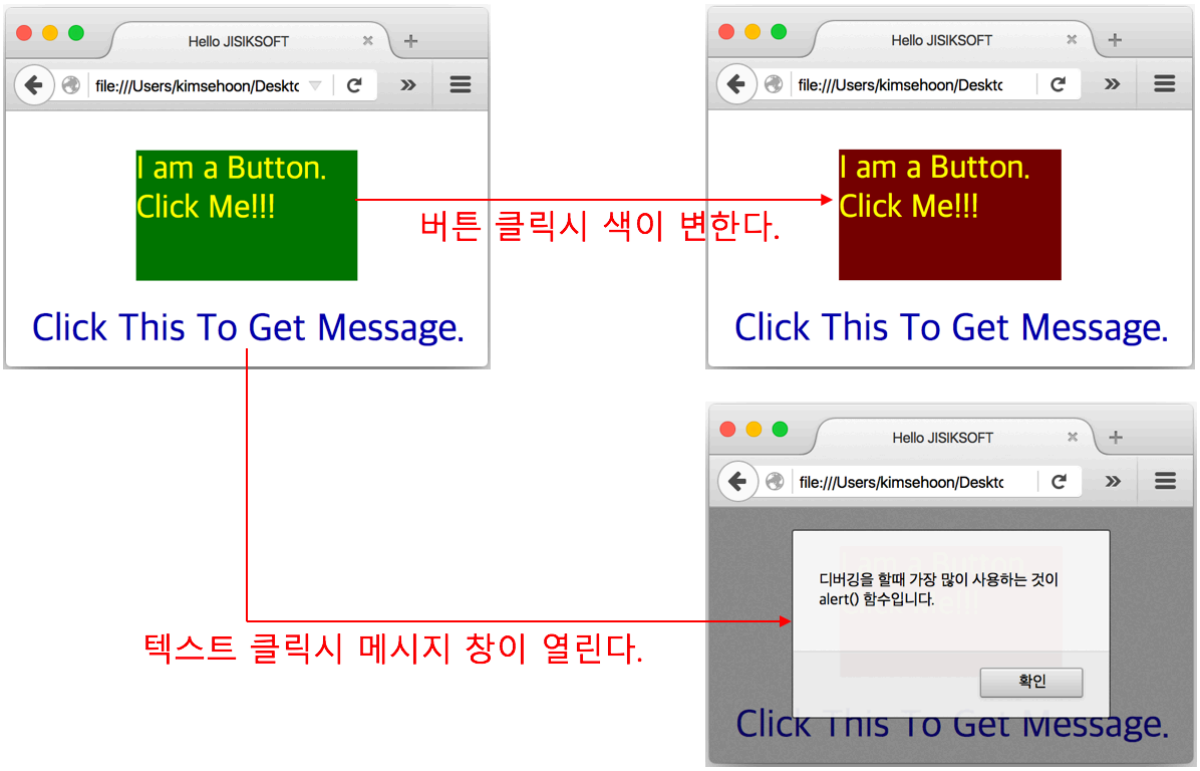
④ 글자가 클릭되면 태그의 onclick 이벤트가 실행되며, onclick 이벤트는 runText() 함수를 실행한다. 이때 alert()이라는 경고창이 발생하는데, JavaScript 프로그래밍을 하면서 가장 많이 사용하는 것이 alert() 함수다. 프로그래머의 가장 절친한 친구는 디버깅Debugging이기 때문에 alert() 함수에게 고맙다.

⑤ 화면에 사각형을 만들어 주며, 태그 안에 onclick 이벤트를 만들었다. 모든 태그에서는 이와 같이 이벤트를 발생시킬 수 있다.

⑥ 글자에도 이벤트를 사용할 수 있으며, 태그를 사용하면 된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.3/1/clickEvent.html

그림 1-41 onclick 이벤트 발생시 JavaScript 함수를 실행한 결과



CSS를 .css 파일로 분리할 수 있듯, JavaScript도 .js 파일로 분리가능하다. HTML 파일에 직접 JavaScript를 사용할 수도 있지만 실제 웹 프로그래밍을 하다보면 아주 많은 JavaScript 함수를 만들어야 하기 때문에 js 파일이 유용하다. 이 책에서도 이후부터는 JavaScript 코드는 js 파일에서 대부분 작성될 것이다.

[코드 1-21]과 [코드 1-22]는 위에서 작성된 [코드 1-20]의 HTML 하나의 파일을 .html과 .js 두 개로 분리한 것이며, CSS와 마찬가지로 'clickEvent.js' 파일을 <head></head> 태그에 선언했다.

[코드 1-21] JavaScript 함수를 clickEvent.js 파일로 옮긴 후 파일을 불러온다. (/2/clickEvent.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./clickEvent.css"></link>
        <script src="./clickEvent.js"></script>                    //---①
    </head>

    <body>
        <div id="btn_1" onclick="runButton();"> I am a Button. Click Me!!! </div>
        <span id="text_1" onclick="runText();"> Click This To Get Message. </span>
    </body>
</html>
```

① 모든 프로그래밍 언어는 라이브러리Library나 다른 파일에 있는 코드를 불러와서 사용하는데, HTML에서 JavaScript 파일을 불러오는 방법은 <script></script> 태그의 'src' 속성을 사용하는 것이다. 실제 .html 파일에는 JavaScript 코드가 없지만 브라우저에서는 HTML 파일과 JavaScript 함수가 함께 존재하는 것으로 인식한다.

[코드 1-22] JavaScript 코드는 js 확장자 파일로 옮겨졌다. (/2/clickEvent.js)

```
var arrayColor = [ '#770000', '#007700', '#000077' ];
var index = 1;

function runButton() {
    index = (index + 1) % 3;
    document.getElementById("btn_1").style.backgroundColor = arrayColor[index];
}

function runText() {
```

```
    alert("디버깅을 할때 가장 많이 사용하는 것이 alert() 함수입니다.");
}
```

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.3/2/clickEvent.html

그림 1-42 JavaScript 코드를 js 파일에서 관리해도 같은 결과를 얻는다.

JavaScript의 특징 중의 하나는 함수를 선언하는 방법이다.

[그림 1-43]은 JavaScript에서 함수를 선언하는 두 가지 방법을 보여주는데, 함수를 정의하는 'function'을 함수 이름 앞에서 사용하는 일반적인 방법과 함께 뒤에서도 사용할 수 있는 방법이 있다.

[코드 1-23]은 [코드 1-22]와 같은 동작을 하는 JavaScript 코드인데, 함수를 선언하는 방법만 변경했다. 이와 같이, 'function'을 뒤에서 사용하는 방법은 JavaScript 만의 특징이며, jQuery와 같은 JavaScript 라이브러리를 만들거나 클래스(OOP: Object Oriented Proramming) 형태의 프로그래밍을 구현하는데 유용하게 사용된다. 어느 방법이 더 좋은 방법이라고 판단할 수 없기 때문에, 이 책에서는 두 가지 함수의 선언 방법을 혼용해서 사용한다.

그림 1-43 JavaScript 에서 함수를 선언하는 방법 2 가지

```
<함수 선언 1>
function runButton() {                  //function을 앞에서 선언
        alert("test");
}
<함수 선언 2>
runButton = function() {                //함수명을 function으로 선언
        alert("test");
}
//위의 두가지 방법의 모두 많이 사용된다.
```

[코드 1-23] JavaScript 함수를 정의하는 또 다른 방법 (/3/clickEvent.js)

```
var arrayColor = [ '#770000', '#007700', '#000077' ];
var index = 1;

runButton = function() {
	index = (index + 1) % 3;
	document.getElementById("btn_1").style.backgroundColor = arrayColor[index];
}

runText = function() {
	alert("디버깅을 할때 가장 많이 사용하는 것이 alert() 함수입니다.");
}
```

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.3/3/clickEvent.html

HTML과 JavaScript는 참 재미있는 스트링 사용법이 있는데, 큰따옴표(")와 작은따옴표(')의 기능이 같다는 것이다. 스크립트Script 언어이기 때문에 텍스트 안에 큰따옴표나 작은따옴표를 넣어서 사용할 수 있는데, 이를 위해서는 큰따옴표나 작은따옴표가 같은 기능을 해야만 한다.

[그림 1-44]는 변수에 문자열String을 정의하는 방법이다. 세 번째 예제를 보면 텍스트의 내용 안에 작은따옴표가 포함된 텍스트를 만드는 것이고, 네 번째는 텍스트의 내용 안에 큰따옴표를 사용하기 위해서 사용된 예제이다. 이와 같이, 두 개의 따옴표를 혼용하여 사용하는 이유는 이벤트를 정의할 때 따옴표를 사용하는데, 이벤트를 진행하는 함수 안에 또다른 문자열String을 정의하는 경우가 있기 때문이다. 예를 들면, [코드 1-24]에서 onclick을 정의할 때 함수 안에 매개변수로 문자열String을 넣어주는데, 이것을 가능하게 해주는 방법이 두 개의 따옴표를 사용하는 것이다.

그림 1-44 문자열(String) 정의는 큰따옴표 또는 작은따옴표로 표현한다.

```
<문자열(String) 사용법>
var str = "Hello JisikSoft";        // Hello JisikSoft
var str = 'Hello JisikSoft';        // Hello JisikSoft
var str = "'Hello' JisikSoft";      // 'Hello' JisikSoft
var str = 'Hello "JisikSoft"';      // Hello "JisikSoft"
```

JavaScript 함수의 중요한 특징 중의 하나는 함수를 통하여 전달되는 매개변수Parameter의 생략이 가능한 것이다. JavaScript는 매개변수를 가지고 다양한

기능을 하는 함수를 만드는 것이 가능하다.

[코드 1-24]는 하나의 함수를 호출하는데 매개변수의 갯수가 다양하게 전달되는 코드를 보여준다. 4개의 onclick이 실행될 때, 단지 하나의 함수 "runFunction()" 함수가 실행된다. 그런데, "runFunction()" 함수를 호출하는 onclick에서의 실행되는 함수는 매개변수의 갯수가 일정하지가 않고 생략된 경우들이 있음을 보여준다. 일반적인 프로그래밍 언어에만 익숙했던 필자가 JavaScript 코드를 보면서 혼동이 왔던 부분 중의 하나가 매개변수의 생략이 가능하다는 특징이었다. [코드 1-25]는 "runFunction()" 함수를 정의한 것을 보여주는데, 매개변수가 5개인 함수로 정의되었다. [코드 1-24]에서 주의 깊게 봐야할 것은 문자열String을 만드는 큰따옴표(")와 작은따옴표(')의 사용인데, 첫 번째 와 나머지 에서 사용된 따옴표들이 다름을 알수 있다. 이렇게 따옴표를 다르게 사용했음에도 onclick 이벤트는 문제없이 실행되며, 위에서 설명한 문자열String의 사용법을 쉽게 이해할 수 있으며 아주 중요하다.

[코드 1-24] JavaScript 의 특징 중 하나는 함수의 매개변수가 생략이 가능 (/4/parameter.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./parameter.css"></link>
        <script src="./parameter.js"></script>
    </head>
    <body>
        <div id='text_1' onclick='runFunction("text_1","Hello","지식소프트","웹","프로그래밍");'>
                click_1 </div>                                                    //---①
        <div id="text_2" onclick="runFunction('text_2','Hello','지식소프트','웹');"> click_2 </div>
        <div id="text_3" onclick="runFunction('text_3','Hello','지식소프트');"> click_3 </div>
        <div id="text_4" onclick="runFunction('text_4','Hello');"> click_4 </div>     //---②
    </body>
</html>
```

① 문자열String은 작은따옴표(')로 정의되며, onclick의 값으로 넣어진 runFunction()함수의 매개변수로 문자열String이 사용되기 때문에 큰따옴표(")를 사용하였다. 즉, onclick에서의 큰따옴표(")는 단지 문자열의 일부분일 뿐이다.

② 가장 처음에 정의된 문자열과 반대로 큰따옴표(")로 문자열을 정의했으며, onclick

에 의해서 실행되는 runFunction() 함수의 매개변수는 단지 두 개일 뿐이지만 함수는 정상적으로 동작한다.

[코드 1-25] 생략이 가능한 매개변수의 특징을 가진 JavaScript 함수 (/4/parameter.js)

```
runFunction = function(para_1, para_2, para_3, para_4, para_5) {                    //---①

    var str = "";                                                                   //---②

    if (para_1 == null)                                                             //---③
        return;
    if (para_2 != null)                                                             //---④
        str += ' ' + para_2;
    if (para_3 != null)
        str += ' ' + para_3;
    if (para_4 != null)
        str += ' ' + para_4;
    if (para_5 != null)
        str += ' ' + para_5;

    document.getElementById(para_1).innerHTML = str;                                //---⑤
}
```

① runFunction() 함수는 다섯 개의 매개변수를 갖는다. 매개변수의 숫자가 5개 미만이어도 함수의 실행은 가능하다.

② 문자열 변수 str을 빈 문자열로 초기화했다.

③ 'para_1'은 [코드 1-24]에서 태그의 id 값을 넘겨 받으며 함수의 마지막에 id 값을 갖은 객체를 가져오기 위해서 사용된다. 여기서는 객체를 받기 위한 para_1의 값이 없으면 함수를 종료한다. 즉, 매개변수가 없는 함수의 실행은 가능하지만, 이 함수의 특성상 첫 번째 매개변수 조차도 없으면 이후의 진행과정이 무의미하기 때문에 return으로 함수를 종료한다.

④ 매개변수가 두 번째 이후로 존재하면 공백과 함께 매개변수를 문자열 변수 str에 계속 추가한다.

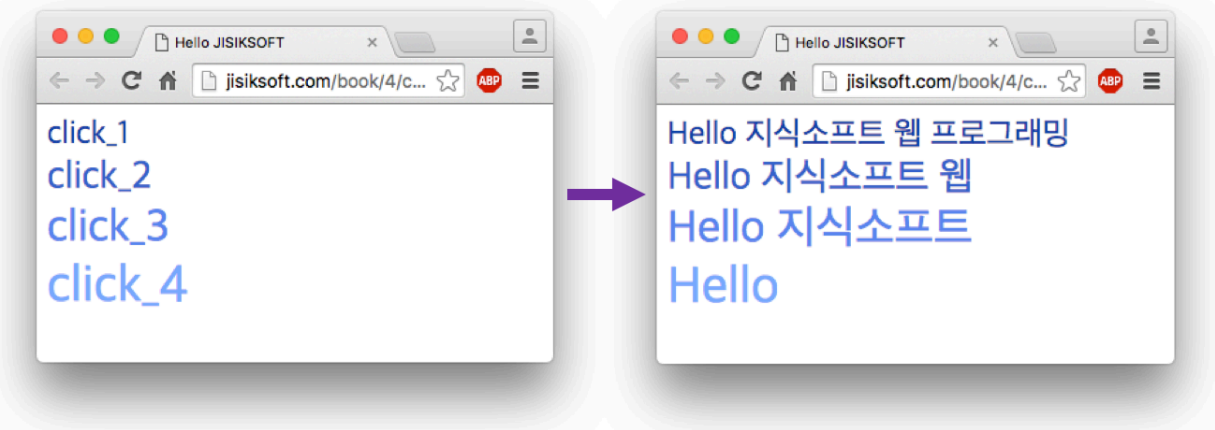
⑤ [그림 1-40]에서 설명한 내용으로 첫 번째 매개변수로 받은 para_1의 id 값을 가진 객체를 "document.getElementById(para_1)"으로 가져와서, 해당 객체의 내용을 설정하되 문자열 str 변수의 값으로 변경한다.

[그림 1-45]는 지금까지 구현한 함수의 결과를 브라우저에서 보여주는데,

문자열들을 모두 클릭하면 오른쪽과 같은 결과를 얻게 된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.3/4/parameter.html

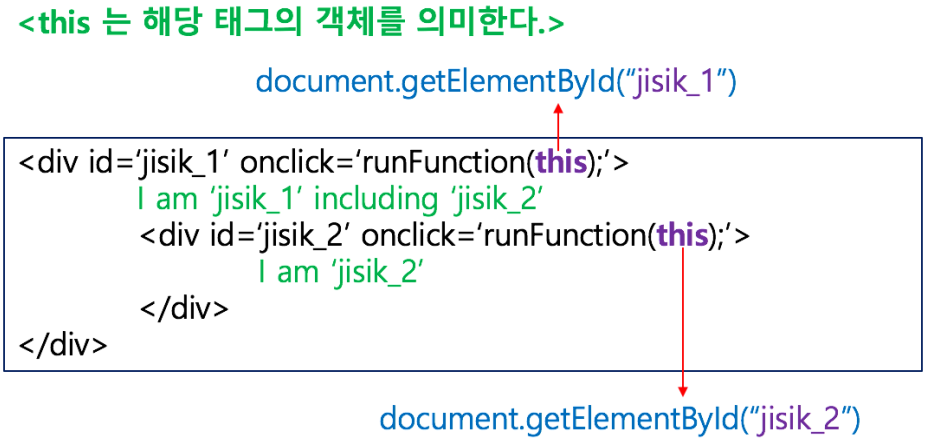
그림 1-45 매개변수가 줄어들면서 함수가 호출되어도 문제없이 동작한 결과



이번 장에서 DOM의 구조에 대해서 설명을 하였는데, 웹 프로그래밍에서 많이 사용하는 것 중의 하나가 'this'라는 예약어이다. 'this'는 해당 태그의 객체를 의미한다.

[그림 1-46]을 보게되면 onclick의 이벤트에서 실행되는 함수 runFunction()의 매개변수로 'this'가 사용되었다. 이와 같이 JavaScript를 호출할 때 'this'가 사용되는데, 이것은 'this'가 사용된 태그의 객체를 의미한다. 즉, 첫 번째 this는 id 값이 'jisik_1'인 객체를 의미하고, 두 번째 this는 id 값이 'jisik_2'인 객체를 의미한다. 'this'라는 예약어를 사용해서 객체 자체를 다른 곳으로 전달할 수 있는데, Java나 JavaScript에서는 사용하지 않지만, 'C/C++' 프로그래밍 관점에서는 해당 객체를 가리키고 있는 포인터Pointer를 넘겨주는 'Call-By-Reference'의 개념으로 이해하면 된다. ("How-to" Series" 첫 번째 "Big Number 연산"을 읽은 독자를 위해서 설명하였다.)

그림 1-46 this 는 this 가 명시된 곳의 객체를 의미한다.



[코드 1-26]과 [코드 1-27]은 'this'를 사용해서 함수를 구현하는 방법을 보여준다. [코드 1-24]와 [코드 1-25]에서는 객체의 id 값을 매개변수로 넘겨주고, JavaScript 함수에서 id 값을 사용해서 객체에 접근해서 객체의 내용을 바꾸었다. 하지만, [코드 1-26]와 [코드 1-27]에서는 객체를 넘겨주는 것이기 때문에 id 값을 사용해서 객체를 찾는 불필요한 작업을 하지 않아도 된다. 사용자는 객체를 넘겨주었을 때의 차이를 느끼지는 못하지만, 프로그래머의 관점에서 생각하면 'this'의 사용이 속도 측면에서 더 효과적이라 볼 수 있다.

[코드 1-26] JavaScript 의 특징 중 하나는 함수의 매개변수가 생략이 가능 (/5/parameter.html)

```
<!DOCYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./parameter.css"></link>
        <script src="./parameter.js"></script>
    </head>

    <body>
        <div id='text_1' onclick='runFunction(this,"Hello","지식소프트","웹","프로그래밍");'>
                click_1 </div>                                                    //---①
        <div id="text_2" onclick="runFunction(this,'Hello','지식소프트','웹');"> click_2 </div>
        <div id="text_3" onclick="runFunction(this,'Hello','지식소프트');"> click_3 </div>
        <div id="text_4" onclick="runFunction(this,'Hello');"> click_4 </div>
    </body>
</html>
```

① runFunction() 함수의 첫 번째 매개변수로 this를 사용하였으며, runFunction() 함수에서 id 값이 'text_1'인 태그의 객체를 핸들링Handling하게 된다. 모든 태그들이 id 값을 가지고 있지만, this를 사용하게 되면 사실상 id 값이 존재하지 않아도 된다. 여기서 id 값들이 설정된 이유는 CSS 디자인 변경만을 위해서 사용되었다.

[코드 1-27] 매개변수로 객체(object)를 받아서 사용이 가능하다. (/5/parameter.js)

```
runFunction = function(object, para_2, para_3, para_4, para_5) {          //---①

    var str = "";
```

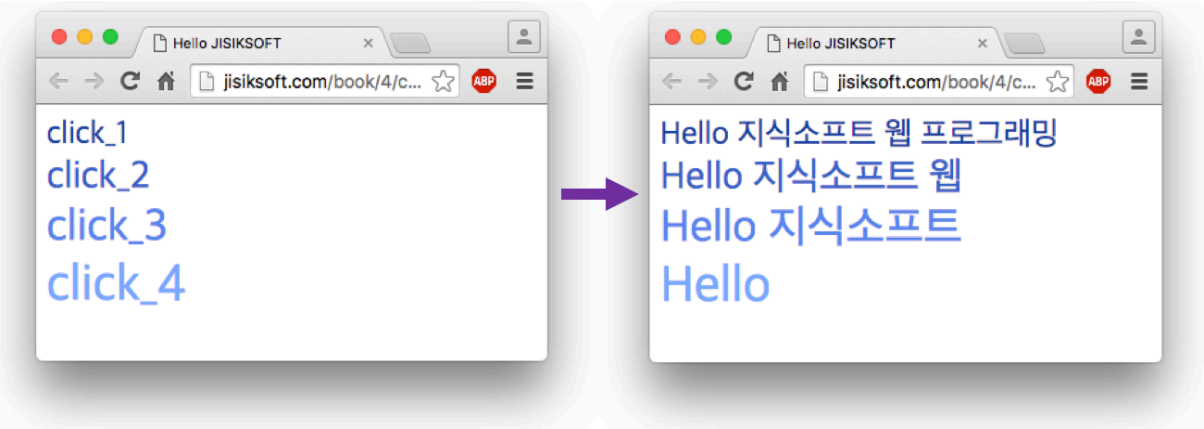
```
    if (object == null)
        return;
    if (para_2 != null)
        str += ' ' + para_2;
    if (para_3 != null)
        str += ' ' + para_3;
    if (para_4 != null)
        str += ' ' + para_4;
    if (para_5 != null)
        str += ' ' + para_5;

    object.innerHTML = str;                                    //---②
}
```

① 함수의 첫 번째 매개변수로 객체를 받는다. [코드 1-25]에서와 같이 'para_1'으로 매개변수 이름을 사용해도 상관은 없지만, 매개변수도 이름을 정할 때에는 특징을 알수 있게 하는 것이 좋다.

② object는 객체이기 때문에 객체의 내용을 변경할 수 있는 innerHTML 속성Property을 직접 사용할 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.3/5/parameter.html

그림 1-47 태그의 id 를 사용하지 않고 this 를 함수의 매개변수로 사용해도 같은 결과를 보여준다.



1990년대부터 프로그래밍 언어는 클래스Class 개념(OOP: Object-Oriented Programming)으로 변화되어 왔다. (여기서 말하는 클래스는 CSS 디자인 속성에서 그룹으로 나누기 위해서 사용되는 클래스와는 개념이 다르다. HTML 언어에서 태그들의 그룹을 나누기 위해서 사용되는 클래스들은 이름을 값으로 가지지만, 여기서 말하는 클래스는 프로그래밍 언어에서 사용하는 추상적인 개념이다.) JavaScript는 클래스에서 출발하지 않았기 때문에 JavaScript를 많이 사용하는

프로그래머들도 클래스를 사용할 수 있다는 사실을 모르는 경우가 있다. JavaScript 언어도 다른 언어와 마찬가지로 클래스를 만들어서 사용할 수 있는데, JavaScript가 얼마나 뛰어난 언어인지를 느낄 수 있을 것이다. 엄밀히 말하자면 JavaScript는 클래스 언어가 아니지만, 클래스의 기본 구조에 맞게 구현이 가능하다고 볼 수 있다. 이렇게 클래스로 만들 수 있다는 것은 다양한 기능을 하는 클래스들을 만들어서 라이브러리Library 형태로 관리하면서 코드의 재사용이 용이하게 하는 일반적인 프로그래밍 언어와 같이 사용할 수 있다는 것이다.

그림 1-48 JavaScript 언어에서의 클래스와 사용 방법

<Class 구조>

```
var ClassName = function( ) {
        this.variable_1 = "value";
        this.variable_2 = "value";
}

ClassName.prototype = {
        fixedVariable_1 : "value",

        method_1 : function( ) {
        },
        method_2 : function( ) {
        }
}
```

<Class 생성 및 사용>

```
var jisiksoft = new ClassName();

jisiksoft.method_1( );
jisiksoft.method_2( );
```

[그림 1-48]은 JavaScript에서의 클래스의 정의와 사용방법을 보여주는데, 클래스 이름 앞에 'var'를 사용해서 변수로 선언하면서 'function()'을 사용해서 함수로 지정하였다. 클래스의 기본 개념은 클래스의 변수로 사용되는 속성Property과 함수로 사용되는 메서드Method 두가지를 이해해야 하는데, 속성들은 클래스 함수 안에 정의되고 메서드들은 prototype에 정의하게 된다. 프로그래밍 언어를 좀더 깊이 있게 이해하기 위해서는 컴퓨터 메모리(Memory: RAM)와의 상관관계를 이해하는 것이 좋으며, 클래스가 생성된다는 것은 메모리에 클래스의 공간이 만들어지는 것을 의미한다. [그림 1-48]의 오른쪽에서 클래스의 생성을 위해서 'new'를 사용하였는데, 클래스는 함수이기 때문에 "new ClassName()"과 같이 선언되었다. 'new'를 사용해서 클래스를 생성하게 되면 클래스가 메모리의 일정공간에 만들어지게 되는데, 클래스의 prototype은 메모리의 공간을 차지하지 않고 클래스의 속성값들을 가지고 있는 클래스 변수들만이 메모리에 올라가게 된다. [그림 1-48]의 클래스 구조에서

보면 클래스는 다수가 'new'를 사용하여 생성할 수 있는데, 'this.variable_1'과 'this.variable_2'와 같은 속성들은 해당 클래스에 종속되어 다수가 생성될 수 있다. 그러나, 아래의 prototype에 있는 내용들은 중복되어서 메모리에 생성되지 않는다. 즉, prototype의 'fixedVariable_1'은 클래스가 계속 생성되어도 하나만 존재하며 클래스에서 변경하지 않는 값을 사용하는 것이 좋다. 클래스에서 선언되는 변수(속성Property)와 함수(메서드Method)들을 클래스 안에서 사용할 때는 'this'를 앞에 붙여서 사용하는데 웹 프로그래밍에서 사용하는 'this'는 항상 객체를 의미한다. 즉, HTML 언어에서의 this는 해당 태그 객체를 의미한다면, 클래스 안에서 사용되는 this는 클래스 객체라고 보면 된다. 고급 웹 프로그래밍을 하기 위해서는 this의 개념을 잘 이해하는 것이 좋다. 클래스의 prototype에서 ':'(콜론)과 ','(쉼표)의 사용이 중요한데, prototype에서는 변수와 함수들이 이름으로만 정의되며, ':'(콜론)은 '='(Equal)과 같은 개면이며 각각의 정의된 이름들의 구분을 위해서 ','(쉼표)만을 사용할 수 있다. 프로그래밍 언어라는 것은 해당 언어의 정해진 규칙을 잘 이해하고 그 언어의 틀 안에서 코드를 정확히 구현해야 한다.

[코드 1-28]은 Book이라는 클래스를 만든 것이며, 책이 출판된 연도와 책 제목을 클래스 변수Property에 저장한다. Book 클래스가 생성될 때 매개변수Parameter로 태그 객체Tag Object, 연도, 책 제목을 받아서 Book 클래스의 변수에 저장하고, 마지막에 prototype에서 정의된 setBook() 함수Method를 실행한다. 'setBook()' 함수와 같이 클래스의 함수Method는 클래스가 생성될 때 실행할 수 있는데, 클래스가 생성될 때 초기화해야 하는 항목들이 있을 때 이와 같은 방법을 사용한다. Book 클래스의 prototype에 있는 color는 세 개의 RGB 값을 저장하고 있는 배열이며, 이 값은 클래스가 생성되고 실행되는 동안 변하지 않는 데이터이기 때문에 prototype 안에서 정의되었다. 모든 클래스의 함수Method들은 prototype 안에서 선언되는데, setBook 함수는 클래스가 생성될 때 매개변수로 받은 객체Object의 디자인과 내용을 변경하는 동작을 수행하며, getBookInfo는 해당 함수가 실행될 때 매개변수로 받은 object 객체의 디자인과 내용을 변경한다. 코드를 보게되면 setBook함수와 getBookInfo에서 사용된 object 변수는 별개인데, this.object는 클래스가 생성될 때 저장된 객체를 의미하며 getBookInfo 함수의 object는 클래스 객체를 의미하는 this를 앞에서 선언하지 않았기 때문에 함수가 실행될 때 매개변수로 넘겨받은 객체이다. 'this'가 사용된 것과 사용되지 않은 것의 차이를 명확히 이해해야

클래스의 개념이 명확해지며 구현이 좀더 쉬워진다.

[코드 1-29]는 HTML 코드에서 Book 클래스를 생성하고 클래스의 함수Method를 실행하는 것을 보여주는데, onclick 이벤트와 같이 모든 이벤트에서는 JavaScript의 코드를 직접 넣어서 코드를 구현할 수 있다. 즉, 많은 기능을 수행하는 코드는 JavaScript 함수를 만들어서 사용하는 것이 좋지만, 간단한 JavaScript 코드는 HTML 태그에 직접 넣어서 구현하는 것도 가능하다.

[코드 1-28] JavaScript 클래스 - Book 클래스 (/6/classJS.js)

```
var index = 0;                                                          //---①

var Book = function(object, year, title) {                              //---②

    this.object = object;                                               //---③
    this.year = year;
    this.title = title;
    this.index = index++;                                               //---④

    this.setBook();                                                     //---⑤
}

Book.prototype = {                                                      //---⑥

    color : [ '#369ead', '#c24642', '#7f6084' ],                        //---⑦

    setBook : function() {                                              //---⑧
      this.object.style.backgroundColor = this.color[this.index];
      this.object.style.fontSize = "15px";
      this.object.style.lineHeight = "38px";
      this.object.innerHTML = "How-to Series " + (this.index + 1) + "<br>" + this.year + " ?";
    },

    getBookInfo : function(object) {                                    //---⑨
      object.style.backgroundColor = this.color[this.index];
      object.style.fontSize = "15px";
      object.innerHTML = this.title;
    }
}
```

① 브라우저에서 버튼을 클릭하면 해당 버튼의 바탕색이 다양하게 변경된다. ⑦에는 색의 변경을 위한 세 가지 RGB 값들이 정해져 있으며 index는 전역변수로서 세 가

지 색의 순서Index를 정하기 위해서 사용된다.

② Book 클래스가 생성될 때 세 가지 매개변수Parameter들을 받는데, object는 클래스를 생성한 버튼의 태그 객체Tab Object를 의미하며, 책이 출간된 연도Year와 책 제목Title을 받게된다.

③ this는 Book 클래스 객체Object를 의미하며, this.object는 Book 클래스의 변수Property를 나타내며 object는 매개변수로 받은 객체이다. 'year'와 'title'도 이와 같이 클래스의 변수에 대입해서 메모리에 저장하게 된다.

④ 코드의 가장 위에서 정의된 index 전역변수는 변경되는 색의 순서를 정하기 위해서 사용되는데 0으로 초기화된 index 값은 클래스가 생성될 때마다 1씩 증가하게 된다. 'index++'와 같이 후위연산자로서 '++'가 사용되었는데, index의 값이 this.index에 저장된 이후에 index는 1이 증가된다. 프로그래밍 언어는 많은 규칙들이 공통으로 사용되는데 C/C++, Java, JavaScript와 같은 언어 중 어느 하나만을 깊게 알고 있으면 다른 언어를 사용해서 프로그래밍을 하기 쉬워진다. 프로그래밍 언어에 자유롭다는 것은 어느 한 언어에 대한 깊은 지식을 가지고 다른 언어를 쉽게 접근할 수 있는 것이다. 쉬운 예가 여기서 사용하는 후위연산자 '++'의 사용은 대부분의 언어에서 공통적으로 사용되는 연산자이다.

⑤ Book 클래스가 생성될 때, 자동적으로 실행되는 함수가 'setBook()' 함수인데, prototype에 정의된 'setBook()' 함수는 클래스를 생성한 태그 객체(this.object)의 디자인과 내용을 변경한다.

⑥ 클래스의 prototype은 클래스가 생성될 때마다 메모리에 올라가는 것이 아니라, 메모리의 한 영역에 고정되어서 생성된 다수의 클래스들이 공통적으로 사용하는 변수와 함수들을 정의한다.

⑦ Book 클래스에서 공통적으로 사용되는 변수가 color 변수이며 배열Array로 정의되었다. 'color'는 세 가지 RGB 값을 갖는데 예제 코드에서 생성되는 세 가지 클래스에 하나씩 사용한다.

⑧ this는 해당 Book 클래스의 객체를 의미하며, this.object는 태그 객체이기 때문에 HTML 언어에서 사용하는 style 속성을 이용해서 backgroundColor(바탕색)을 변경하게 된다. 'this.color'는 Book.prototype에서 선언된 color 변수를 의미하며, this.index는 생성된 Book 클래스의 index를 의미한다. 전체 코드에서 전역변수 index와 Book 클래스의 index는 같은 이름을 갖지만, this의 사용으로 this.index는 Book 클래스 안에서의 변수Property가 된다.

⑨ Book 클래스가 선언된 이후에 실행할 수 있는 getBookInfo() 함수는 매개변수로 받은 object라는 태그 객체Tag Object의 디자인과 내용을 변경한다. 함수 안에서 사용된 object가 this를 앞에 선언하지 않았기 때문에 매개변수임을 의미하며, object는 태크 객체이기 때문에 style 속성의 사용이 가능하다.

[코드 1-29] 클래스는 new 를 사용해서 생성하고 클래스의 함수(Method)를 사용한다. (/6/classJS.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./classJS.css"></link>
        <script src="./classJS.js"></script>

        <script>
                var book_1, book_2, book_3;                                    //---①
        </script>
    </head>

    <body>
     <div class="area">
        <div class="btn_1" onclick="book_1=new Book(this,'2013','Big Number 연산');">
                                        Click 1</div>                          //---②
        <div class="btn_2" onclick="if (book_1 != null) { book_1.getBookInfo(this); }">
                                        Click 6</div>                          //---③
     </div><br>
     <div class="area">
       <div class="btn_1" onclick="book_2=new Book(this,'2014','주식분석프로그램만들기');">
                                        Click 2</div>
        <div class="btn_2" onclick="if (book_2 != null) { book_2.getBookInfo(this); }">
                                        Click 5</div>
     </div><br>
     <div class="area">
        <div class="btn_1" onclick="book_3=new Book(this,'2015','OCR 프로그래밍');">
                                        Click 3</div>
        <div class="btn_2" onclick="if (book_3 != null) { book_3.getBookInfo(this); }">
                                        Click 4</div>
     </div>
    </body>
</html>
```

① 세 개의 클래스가 생성되는 코드이며, 전역변수로 선언되 세 개의 변수는 생성되는 클래스들의 객체를 가리키는 변수들이다. 생성되는 클래스들은 이와 같이 변수들

을 이용해서 해당 클래스들을 구분하며, 해당 클래스의 함수들을 개별적으로 실행할 수 있다.

② 이벤트에서는 JavaScript 언어를 직접 사용할 수 있는데, 간단한 코드는 함수를 추가적으로 만들지 않고 JavaScript 언어를 사용하기도 한다. Book 클래스는 new를 사용해서 생성되며, 첫 번째 매개변수인 this는 HTML 태그 안에서 사용되었기 때문에 태그 객체Tag Object를 의미한다.

③ 생성된 Book 클래스의 함수Method를 실행하는 코드이며, "if (book_1 != null) { }"의 의미는 book_1 클래스가 생성되지 않았다면 아무런 동작도 하지 않기 위해서 사용한 것이다. 즉, book_1 클래스가 생성되었다면 해당 클래스의 getBookInfo() 함수를 실행한다. 여기서 사용된 this는 해당 이벤트를 발생시킨 주체인 태그 객체를 의미한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.3/6/classJS.html

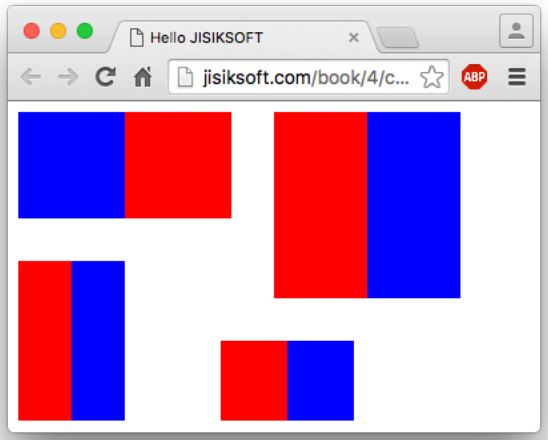
그림 1-49 버튼을 순서대로 누르면 3 개의 클래스가 생성되며, 오른쪽과 같은 결과를 얻게 된다.

클래스Class는 하나만 만들어 놓으면 필요할 때마다 쉽게 사용이 가능하며, JavaScript 라이브러리Library를 만들어서 일정 금액의 사용료를 받고 판매하는 회사들은 특정 동작을 수행하는 JavaScript 클래스를 만드는 것이다. 이번에는 경찰 사이렌 기능을 하는 클래스를 만들어서 클래스 사용의 예를 한번 더 확인해 보고자 한다. [그림 1-50]은 붉은색과 파란색이 서로 바뀌면서 경찰 사이렌 효과의 결과를 보여주는데

변하는 시간은 4개의 사이렌이 저마다 다르며, 크기는 사각형에서 설정된 가로(width)와 세로(height)로 정할 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.3/7/police.html

그림 1-50 하나의 클래스로 다수의 사이렌을 만들어 실행한 결과



사이렌을 만드는 것은 사각형을 두 개로 나누어서 바탕색을 동시에 변경해주면 된다. 색을 변경하는 타이밍Timing도 프로그래머가 정하면 되는데, 대부분의 코드에서 시간설정은 1/1000초 단위로 구현된다. (ex: 1=1ms, 1000=1초) [그림 1-51]은 두 개의 색이 번갈아가며 변경되는 그림을 보여주며, 이것이 연속적으로 수행되면서 사이렌 효과를 내게 된다.

그림 1-51 사이렌 효과는 사각형의 색을 동시에 변경해주면 된다.

코드의 timeChange 값만큼 Delay 후에 색이 반복하여 변한다.

[코드 1-30]은 경찰 사이렌을 구현하는 클래스이며 사격형 태그의 id 값을 container라는 매개변수로 받아서 안에 두 개의 사각형을 추가한 후에 바탕색을 파란색과 붉은색으로 변경하는 작업을 수행한다. PoliceLight 클래스에서 사용되는 함수Method는 setLight()와 turnOnLight() 두 개의 함수들인데,

setLight() 함수는 사각형 안에 두 개의 사각형을 추가하고 turnOnLight()는 일정 간격으로 만들어진 사각형의 바탕색을 변경한다.

[코드 1-30] JavaScript 의 클래스 형태를 보여주는 PoliceLight 클래스 (/7/policeLight.js)

```
var PoliceLight = function(container) {                                    //---①

    this.flagLight = false;                                                //---②
    this.timeChange = null;
    this.leftLight = null;
    this.rightLight = null;

    this.setLight(container);                                              //---③
    this.turnOnLight();
}

PoliceLight.prototype = {                                                  //---④

    twoColor : [ '#0000ff', '#ff0000' ],                                   //---⑤

    setLight : function(container) {                                       //---⑥
        var object = document.getElementById(container);                   //---⑦
        var width = object.style.width;                                    //---⑧
        var height = object.style.height;
        var halfWidth = (width.split('px')[0] / 2) + 'px';                 //---⑨

        var str = '<div id="'+container+'_left" style="width:'+halfWidth+'; \   //---⑩
                    height:'+height+';float:left;"></div> \
                    <div id="'+container+'_right" style="width:'+halfWidth+'; \
                    height:'+height+';float:left;"></div>';
        object.innerHTML = str;                                            //---⑪

        this.timeChange = parseInt(Math.random() * 1000) + 200;            //---⑫

        this.leftLight = document.getElementById(container+'_left');       //---⑬
        this.rightLight = document.getElementById(container+'_right');
    },

    turnOnLight : function() {                                             //---⑭

        if (this.flagLight) {                                              //---⑮
```

```
            this.leftLight.style.backgroundColor = this.twoColor[0];
            this.rightLight.style.backgroundColor = this.twoColor[1];
            this.flagLight = false;
        } else {
            this.leftLight.style.backgroundColor = this.twoColor[1];
            this.rightLight.style.backgroundColor = this.twoColor[0];
            this.flagLight = true;
        }

        var aThis = this;                                                    //---⑯

        setTimeout(function() { aThis.turnOnLight(); }, aThis.timeChange);   //---⑰
    }
}
```

① 사각형을 만드는 태그의 id 값을 container 매개변수로 받는다. JavaScript 언어는 이와같이 id 값을 받아서 해당 태그 객체를 접근할 수 있게 코드를 만든다. 일반적으로 container라고 변수 이름을 짓는 것은 해당 객체에 무엇을 담기 위한 의미로 해석하는 것이 좋다.

② 'flagLight'은 true와 false 값을 갖는데, true일 때는 사이렌의 왼쪽이 붉은색이고 오른쪽이 파란색일 때를 의미한다. 즉, 해당 색이 어떤 상태인지를 확인한 이후에 flagLight 값에 따라서 서로의 색을 변경해주게 구현한다. 'timeChange'는 1/1000초(milli-second) 단위의 값을 갖게 되며, timeChange 시간을 기준으로 사이렌 색이 변경된다. 'leftLight'은 두 개로 나누어진 사각형 중 왼쪽 사각형의 객체를 가리키며, 'rightLight'은 오른쪽 객체를 가리키게 된다.

③ PoliceLight 클래스는 두 개의 함수Method들을 갖는데, 클래스가 생성되면 두 개의 함수를 모두 실행시키게 된다. 이와 같이 코드를 구현하게 되면 클래스가 생성될 때 모든 동작이 자동으로 이루어지게 된다.

④ PoliceLight 클래스가 생성될 때 메모리에 새로운 공간을 만들어서 사용되는 변수들은 위에서 정의되었으며, prototype에서 정의된 변수와 함수들은 클래스가 생성되어도 공통적으로 사용되는 변수와 함수들을 정의한다.

⑤ 'twoColor'는 파란색과 붉은색의 RGB 값을 저장하는 배열이며 모든 생성된 클래스에서 이 값을 공통적으로 사용한다.

⑥ 'setLight()' 함수는 태그의 id 값을 container라는 매개변수로 받아서 해당 태그 객체 안에 두 개의 객체들을 추가한다.

⑦ 'container' 변수에 설정된 값과 같은 id를 가진 객체를 찾아서 object 변수가 가리키게 된다.

⑧ 'object' 변수가 가리키는 사각형 객체의 가로width와 세로height의 크기를 가져온다.

⑨ 큰 사각형의 가로Width 크기를 알기 때문에 이 가로 크기의 절반 크기를 계산해서 'halfWidth' 변수에 넣어준다. 예를 들면 "width = 146px"과 같이 정의 되었다면 "halfWidth = 73px"이라는 값을 갖도록 계산하는 방법을 보여주는데, JavaScript의 문자열String을 다루는 함수 중에 'split()' 함수를 사용하였다. 'width' 값이 '146px'로 정의되었다면 "width.split('px')" 함수는 'px'를 기준으로 문자열을 나누어서 배열에 저장하기 때문에 ['146', ' ']로 정의되게 된다. 그래서 "width.split('px')[0]"은 배열의 첫 번째 값인 '146'을 갖게되며 이것을 2로 나누기 때문에 '73'이라는 값을 갖게 된다. 'halfWidth' 변수는 픽셀Pixel 값을 갖기 때문에 마지막에 'px' 문자열을 더해서 최종적으로 '73px' 하는 값을 갖는다. JavaScript 언어에서의 숫자와 문자열 처리는 다음장 jQuery에서 예제로 자세히 설명되었다.

⑩ 사각형 안에 두 개의 사각형을 추가하기 위하여 두 개의 태그들을 만들어 주는데 각각의 태그의 id 값은 원래의 사각형 id 값에 '_left'와 '_right'를 더해주게 된다. 즉, 만약 원래의 사각형의 id 값이 'light_1'이라면 왼쪽 사각형은 'light_1_left' id 값을 갖게 되고, 오른쪽 사각형은 'light_1_right' 값을 갖게 된다.

⑪ 'object' 변수는 원래의 태그 객체를 의미하며, 이 객체 안에 새롭게 만들어진 두 개의 사각형 태그들을 HTML 언어의 innerHTML 속성을 사용해서 넣어준다. 고급 웹 프로그래밍을 하기 위해서는 태그 객체의 내용들을 자유자재로 변경할 수 있는 방법을 알고 있어야 하며, 그러기 위해서는 이번 장 앞부분에서 설명한 객체의 개념을 명확히 이해해야 한다.

⑫'timeChange' 변수는 사이렌이 변하는 시간(milli-seconds)을 갖게 되며 임의의 수를 'Math.random()' 함수를 사용해서 얻는다. 0.2초보다 작은 시간은 사이렌이 너무 빨리 변하기 때문에 200이라는 수를 더해서 사이렌이 변하는 속도는 0.2초보다 항상 큰 값을 갖게 된다.

⑬ 'container' 값을 id로 가지고 있는 사각형 안에는 좌우에 두 개의 사각형이 존재하는데, 'leftLight' 변수는 왼쪽 사각형의 객체를 가리키고 'rightLight' 변수는 오른쪽 사각형의 객체를 가리킨다.

⑭ 사이렌의 좌우의 색을 변경하는 동작을 수행하며, 설정된 시간을 기준으로 'turnOnLight()' 함수가 계속 실행된다. 이와 같이 색변경을 반복적으로 수행하면서 사용자는 브라우저에서 사이렌 효과를 보게 된다.

⑮ 'flagLight' 변수가 참이면 왼쪽은 붉은색, 오른쪽은 파란색으로 설정되어 있기 때문에, 좌우의 바탕색을 다른 색으로 변경해준다. 색의 변경이 이루어지면 'flagLight'의 값도 변경해서 이후에 어떠한 색으로 설정되었는지를 확인할 수 있다.

⑯ 아래의 'setTimeout()' 함수 안에서 클래스의 함수Method를 실행시키기 위해서 aThis 변수를 사용하는데, JavaScript 언어의 중요한 객체라는 것을 정확히 이해해야 한다. 여기서 사용하는 this는 해당 클래스의 객체를 의미한다. 하지만, 'setTimeout()' 함수는 이 클래스에서 정의된 함수가 아니고, JavaScript 언어에서 정의된 고유 함수이기 때문에 'setTimeout()' 함수 안에서 this를 사용하면 현재 클래스의 객체가 될 수 없다. 그래서 aThis 변수를 사용해서 현재 클래스의 객체를 가리키게 하고, 'setTimeout()' 함수 안에서 aThis 변수가 가리키는 객체의 함수인 'turnOnLight()' 함수를 실행하게 된다. 이와 같이, 고급 웹 프로그래밍을 하기 위해서는 객체라는 것을 확실히 이해할 필요가 있다.

⑰ JavaScript에는 반복적으로 수행되는 동작을 setTimeout() 함수와 setInterval() 함수를 사용하는데, setTimeout() 함수는 함수 안에서 수행되는 JavaScript 코드를 정의된 시간 이후에 1회만 실행하며, setInterval() 함수는 정의된 시간마다 반복적으로 계속해서 실행한다. 여기서는 setTimeout() 함수가 정의된 turnOnLight() 함수를 setTimeout() 함수 안에서 다시 실행하기 때문에 절대로 setInterval() 함수를 사용해서는 안된다. 이 함수 안에서 사용되는 'aThis.timeChange' 값만큼의 대기Delay 후에 설정된 'aThis.turnOnLight()' 함수를 실행한다.

[코드 1-31]은 PoliceLight 클래스를 new로 생성하고 생성된 객체를 arrayPL이라는 배열에 넣어주는 코드인데, 이와 같이 생성된 클래스 객체를 배열에 넣어서 이후에 해당 클래스로 접근이 가능하게 만들어 주어야 한다. 여기서는 클래스를 생성한 후에 다시 접근할 필요가 없지만 대부분의 클래스를 사용하는 웹 프로그래밍에서는 클래스로의 접근이 수시로 이루어진다.

[코드 1-31] 클래스 PoliceLight 를 new 로 생성하는 JavaScript 함수 (/7/police.js)

```
var arrayPL = [ ];                                          //---①

runOneLight = function(container) {                         //---②
    var pl = new PoliceLight(container);                    //---③
    arrayPL.push(pl);
```

```
}

runPoliceLight = function() {                                    //---④
    runOneLight("light_1");
    runOneLight("light_2");
    runOneLight("light_3");
    runOneLight("light_4");
}
```

① 전역변수로 사용된 arrayPL 변수는 배열로 선언되었으며, 이후에 생성되는 PoliceLight 클래스들의 객체를 가리키는 값들이 차례대로 저장되게 된다.

② 태그 객체들의 id 값을 container 매개변수로 받아서 해당 container를 가지고 PoliceLight 클래스를 생성한다.

③ pl 변수는 new로 생성된 PoliceLight 클래스의 객체를 가리키며, 이 값은 배열 Array의 'push()' 함수를 사용해서 배열에 저장된다. 여기서는 클래스가 생성된 이후에 클래스 객체로 접근을 하지는 않지만, 이렇게 코드를 만드는 것이 JavaScript 언어에서 클래스 객체를 관리하는 좋은 방법이다.

④ HTML 파일에서 만들어진 네 개의 태그 객체들의 id 값들을 가지고 'runOneLight()' 함수를 실행해서 네 개의 사이렌 이미지들을 만드는 함수이다.

[코드 1-32]는 onload 이벤트를 사용해서 'runPoliceLight()' 함수를 실행하는 코드이다. 'onload' 이벤트는 웹 프로그래밍의 파일들이 브라우저에 모두 다운받은 이후에 실행되는 이벤트이며, 사용자 입장에서는 브라우저에서 자동적으로 실행되는 함수를 여기서 사용하게 된다. [코드 1-32]에서 네 개의 사각형을 그리기 위해서 네 개의 태그들이 사용되었는데 CSS 디자인을 위해서 class와 id 값들을 선언했다. 그런데 주의해서 봐야 할 것은 태그 안에 style 속성을 이용해서 사각형의 가로width와 세로height의 크기를 직접 선언했다. 'class'와 'id'를 사용해서 CSS에서 크기를 정할 수 있음에도 태그에서 직접 크기를 정한 이후는 CSS는 모든 코드를 받은 이후에 마지막에 브라우저에서 디자인을 만들어 주는데, onload 이벤트는 CSS가 디자인을 정의하기 전에 실행되기 때문에, 만약 사각형들의 크기가 CSS에서 정의되었다면 'runPoliceLight()' 함수가 실행될 때는 사각형의 크기는 설정되지 않은 상태에서 함수가 실행된다. 그래서, 이와 같이 사각형의 크기를 태그 안에서 직접 정의를 해주어야 JavaScript 함수가 실행될 때 해당 사각형의 크기를 알 수 있다.

[코드 1-32] JavaScript 의 클래스 사용을 보여주기 위한 HTML 코드 (/7/police.html)

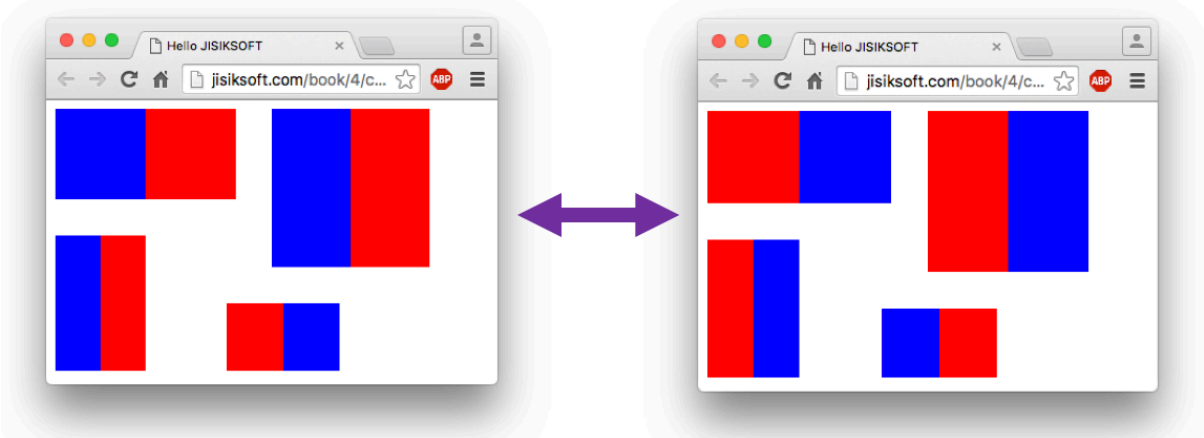
```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./classJS.css"></link>
        <script src="./policeLight.js"></script>
        <script src="./classJS.js"></script>
    </head>

    <body onload="runPoliceLight();">                                    //---①

        <div class="light" id="light_1" style="width:160px;height:80px;"></div>    //---②
        <div class="light" id="light_2" style="width:80px;height:120px;"></div>
        <div class="light" id="light_3" style="width:140px;height:140px;"></div>
        <div class="light" id="light_4" style="width:100px;height:60px;"></div>
    </body>
</html>
```

① 'onload' 이벤트는 body 태그에서 일반적으로 사용하는데, body 태그는 브라우저에서 보여지는 모든 태그 객체를 포함하기 때문에 모든 태그 객체를 받은(load) 이후에 'runPoliceLight()' 함수를 실행한다.

② 모든 디자인을 CSS 파일에서 정의하는 편이지만, 웹 프로그래밍에서는 이와 같이 태그의 style 속성을 직접 넣어주어야 할 때도 있다. CSS 디자인 속성은 브라우저에서 가장 마지막으로 이루어지는 디자인 작업이라는 것을 이해해야 하는데, 'onload' 이벤트의 JavaScript 함수가 브라우저의 CSS 디자인을 적용하기 전에 실행되며 이러한 이유로 여기서는 사각형을 만드는 태그들의 크기를 태그 안에서 직접 정의했다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.3/7/police.html

그림 1-52 PoliceLight 클래스를 사용한 코드의 결과



웹 프로그래밍을 잘한다는 것은 객체의 의미를 이해하고 JavaScript 언어를 얼마나 효과적으로 잘 사용해서 프로그래밍을 하느냐에 달려있다. 모든 프로그래밍 언어들은 공통적인 개념에서 출발해서 자신만의 독특한 특징을 갖고 있는데, 지금까지 설명한 JavaScript의 코드를 이해하는 독자라면 JavaScript 언어의 핵심을 많이 이해할 수 있었을 것이다. 웹 프로그래밍에서 사용되는 언어들이 모두 중요하지만, 그래도 가장 중요한 언어가 JavaScript라는 것을 필자는 강조하고 싶다.

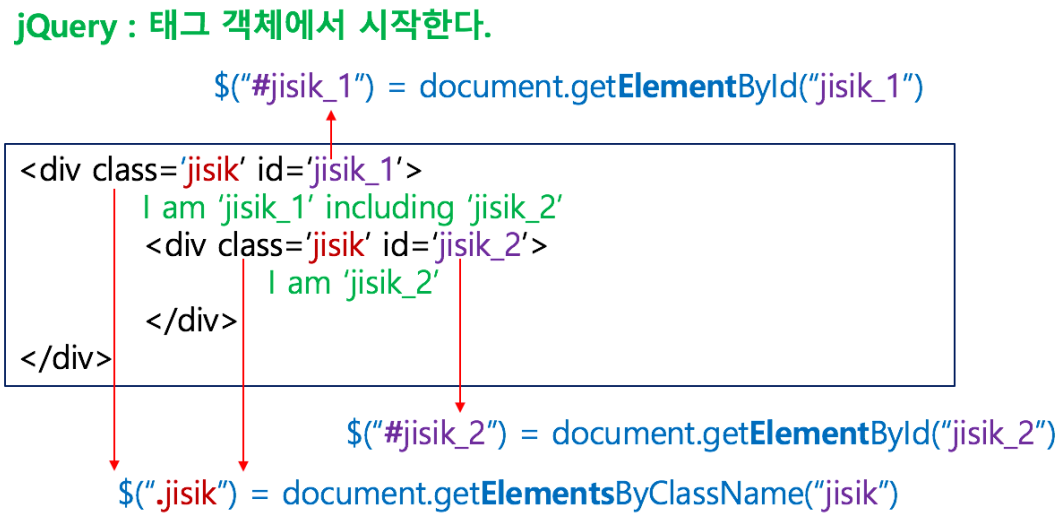
1.4 jQuery

jQuery는 새로운 언어가 아니라 JavaScript로 만들어진 라이브러리Library이다. 현재 웹 프로그래머들이 가장 많이 사용하기 때문에 웹 프로그래밍의 표준으로 사용되고 있다. JavaScript 언어만을 사용해서 웹 프로그래밍을 하게되면 타이핑Typing을 많이 할 수 밖에 없는데, JavaScript 언어의 사용을 간단하게 해주는 역할과 다양한 기능을 함수로 만들어서 제공해 주기 때문에 jQuery를 알면 좀더 빠르고 다양한 프로그래밍 작업을 할 수 있다. 이전 장에서 JavaScript로 클래스를 만드는 방법을 배웠는데, jQuery가 클래스로 만들어지지는 않았지만 우리가 쉽게 사용할 수 있는 무료로 제공되는 클래스 정도로 이해하면 된다. JavaScript는 태그의 객체를 가지고 프로그래밍 작업이 이루어지는데, jQuery는 태그의 객체에서 시작하는 언어라고 이해하면 된다.

[그림 1-53]은 jQuery에서 태그의 클래스나 id 값을 가지고 객체에 접근하는 방법을 보여주는데, JavaScript에서는 항상 document에 접근한 후 객체를 불러오는 함수를 써야 했다면, jQuery는 '$' 문자를 사용해서 객체를 쉽게 가져오게 된다. 프로그래머의 관점에서 본다면 jQuery는 '$()' 함수를 만들어서 사용하는 것이며, 함수 매개변수로 클래스나 id 값을 넣어서 해당 객체를 불러오게 된다. '$()' 함수를 실행해서 결과값으로 갖는 것은 태그 객체이기 때문에 이후에 jQuery의 추가적인 함수들을 사용해서 다양한 프로그래밍 작업이 가능해진다. '$()' 함수의 매개변수로 사용되는 문자열은 CSS에서 사용하는 규칙을 따르는데, 클래스 이름 앞에는 '.'(마침표)를 사용하고 id 이름 앞에는 '#'(샵) 문자를 사용한다. 물론 CSS에서와 같이 태그 이름(ex: div,

span)의 사용도 가능하지만, 실제 웹 프로그래밍에서 사용되는 태그들은 워낙 많기

그림 1-53 jQuery 는 CSS 에서 사용하는 이름 규칙으로 객체에 접근하게 된다.



때문에 클래스와 id 이름만으로 jQuery를 사용하는 것이 좋다. 이전 장에서 웹 프로그래밍에서 가장 중요한 객체에 대하여 설명을 했기 때문에 jQuery가 태그 객체에서 시작한다는 것을 쉽게 이해할 수 있을 것이다.

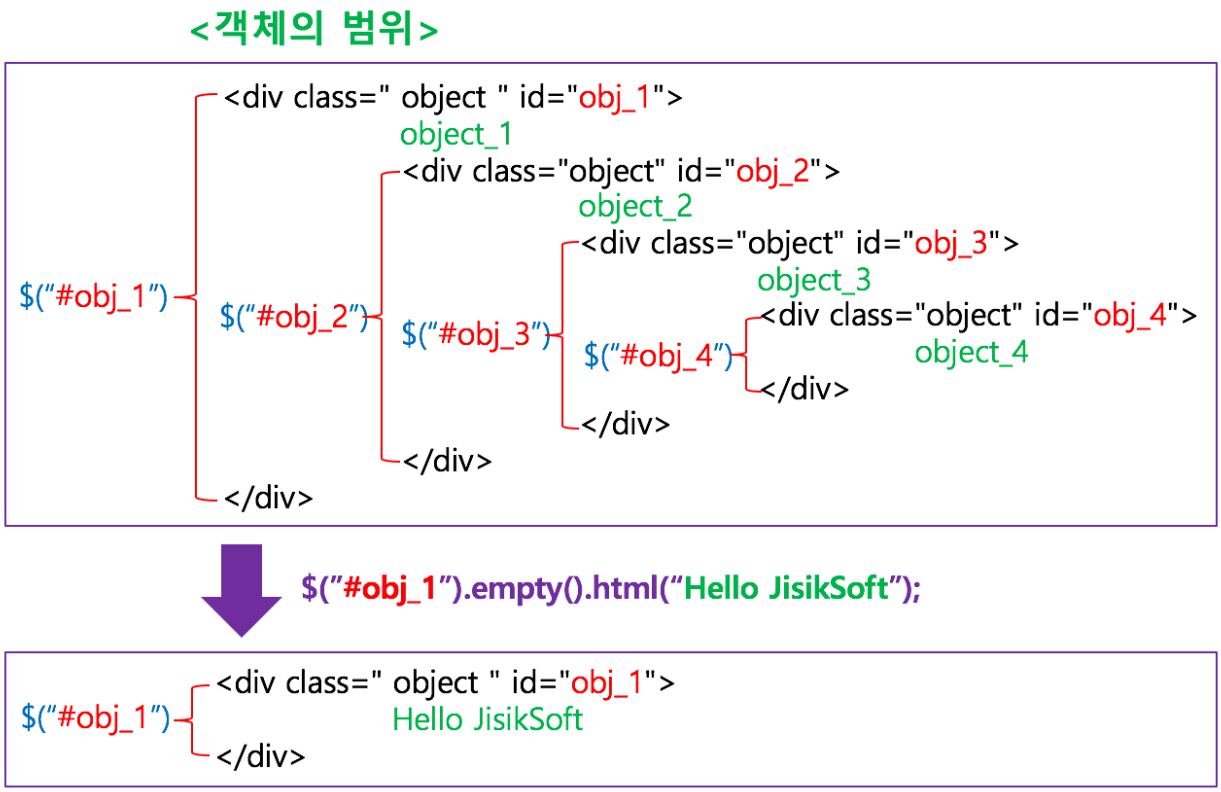
객체에서 시작하는 jQuery는 '$()' 함수 뒤에 사용하기 쉽게 만들어진 jQuery 함수들을 사용해서 필요한 동작들을 구현할 수 있는데, jQuery의 모든 함수들을 외울 필요가 없으며 필요할 때마다 검색(Googling)을 해서 찾아서 사용하면 된다. 아래의 표에는 필자가 가장 많이 사용하는 jQuery 함수의 종류들이다. 함수가 워낙 많으니 jQuery에서 자주 사용하는 몇 개의 함수들만 알고 있는 것이 편하다. JavaScript 언어에서 객체를 가지고 할 수 있는 대부분은 jQuery 함수로 편리하게 만들어져 있다고 이해하면 된다.

jQuery 함수	설명
click()	태그 객체를 마우스 왼쪽 버튼으로 누르면 수행되는 동작을 JavaSript 함수로 구현한다.
hover()	태그 객체 안으로 마우스가 이동하거나 밖으로 나갈 때 수행되는 동작을 구현한다.
css()	태그 객체의 CSS 디자인을 변경할 때 사용한다.
empty()	태그 객체의 내용을 모두 지울 때 사용한다.

html()	태그 객체 안에 내용을 넣을 때 사용한다.
append()	태그 객체 안에 있는 기존 내용을 지우지 않고 매개변수로 전달되는 내용을 뒤에 붙여 넣는다.
width()	태그 객체의 가로Width 크기를 가져오거나 설정할 때 사용한다.
height()	태그 객체의 세로Height 크기를 가져오거나 설정할 때 사용한다.
animate()	태그 객체의 위치, 크기, 디자인 등을 설정된 시간동안 서서히 변화시킬 때 사용하며, 움직이는 동작으로 실행된다.

[그림 1-54]는 객체에 대한 이해를 돕기 위해서 jQuery 함수들을 이용해서 객체의 내용을 변경하는 방법을 보여주는데, jQuery의 함수인 empty()와 html()을 사용해서 가장 상위에 있는 객체의 내용을 변경하였다. 그림에서 위에 있는 네 개의 태그Tag들은 먼저 선언된 것이 부모Parent 객체이고 객체의 안에서 선언된 태그들은 상위 태그의 자식Child 객체다. 첫 번째로 사용된 empty() 함수는 태그 객체 안에 있는 내용을 모두 없애주기 때문에 모든 자식 객체들이 사라지는 것이며, 두 번째 함수인 html() 함수는 태그 객체에 "Hello JisikSoft"라는 문자열String을 추가하기 때문에 그림의 아래와 같은 결과를 얻게 된다. 이미 존재하는 객체들도 프로그래밍 구동하면서 객체를 없앨 수 있는데, 일반적으로 사용하지는 않지만 동적으로 움직이는 고급 웹 프로그래밍을 하기 위해서는 태그 객체를 만들고 지우는 방법을 이해하는 것이 좋다. 예를 들면, 이전 장에서 마지막에 설명한 사이렌 클레스는 클래스가 생성되면서 사각형 안에 두 개의 사각형을 그리기 위해서 두 개의 태그들을 추가했었다. 태그 객체를 중심으로 코드를 해석해서 브라우저에 보여주는 웹 프로그래밍에서는 태그 객체를 추가하고 없애는 일련의 동작들을 수행하는 JavaScript 라이브러리Library들이 많은데, 대표적인 것이 그래프를 만드는 라이브러리들이며 하나의 막대그래프를 그리기 위해서 태그 객체를 추가해주는 방법으로 진행된다. 물론, 현재는 HTML5 언어를 사용해서 그림을 그려주는 그래프 라이브러리들로 변화하고 있지만, 예전에는 태그 객체를 추가하는 방법으로 그래프가 만들어졌다.

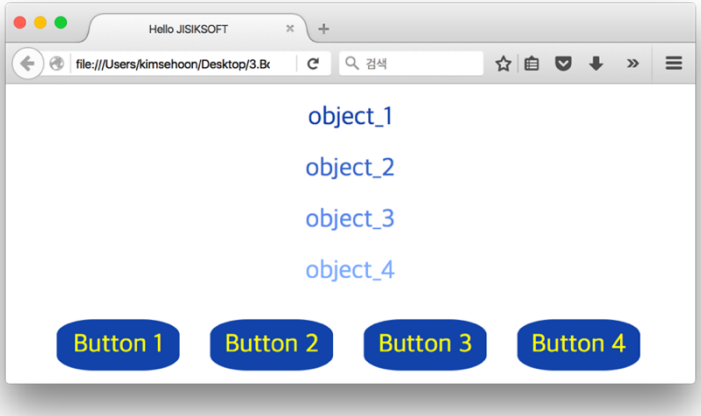
그림 1-54 객체의 범위를 이해하기 위하여 jQuery 의 empty() 함수와 html() 함수를 사용하였다.



이제부터 구현하는 코드는 jQuery가 태그 객체에 접근해서 객체의 디자인과 내용을 변경하는 방법들을 보여주는데, [그림 1-55]는 구현할 내용의 초기화면을 보여준다. 즉, 네 개의 문자열을 보여주는 객체들과 아래 네 개의 버튼들을 화면에서 보여주며, 버튼이 클릭됨에 따라 문자열의 내용과 디자인이 변경되는 것을 확인할 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.4/1/object.html

그림 1-55 jQuery 사용법을 이해하기 위해서 만들어진 코드의 실행 화면



[코드 1-33]은 다수의 태그들을 만들었는데, 네 개는 문자열String를 위

하여 사용되었으며, 나머지 네 개는 버튼 역할을 하는 태그들이다. 주의깊게 봐야 할 것이 문자열String를 위하여 만들어진 태그들인데, 네 개의 태그들이 서로 종속관계를 갖는다. 즉, id 값이 'object_3'인 태그의 내용을 지우면 'object_4'의 태그가 없어지는데, 이와 같이 각각의 태그들은 자신보다 상위에 있는 태그에 의해서 없어지게 된다. 모든 태그들은 클래스와 id 값을 갖는데, CSS 디자인을 위해서도 사용되지만, jQuery에서 객체에 접근하기 위해서도 사용된다. 필자가 가장 좋아하는 HTML 언어에서의 태그는 사각형을 만들수 있는 태그인데, 웹 프로그래밍을 하면서 대부분의 코드를 태그만을 사용해서 구현이 가능하기 때문이다. 브라우저는 사각형이고 사각형을 의미하는 태그로 위치와 크기를 정해서 대부분의 디자인을 할 수 있기 때문이며, 기억력이 나쁜 필자와 같은 프로그래머는 알고 있어야 하는 내용이 적을수록 프로그래밍이 편하다. 프로그래밍을 한다는 것은 모든 것을 기억하고 만드는 것이 아니라 가장 중요한 것을 먼저 이해한 후에 필요한 것을 찾아서 응용하는 것이 가장 효과적임을 강조하고 싶다.

[코드 1-33] jQuery 의 다양한 동작을 표현하기 위해서 만들어진 HTML 코드 (/1/object.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./css/object.css"></link>
        <script src="./js/jquery-2.1.3.min.js"></script>                    //---①
    </head>
    <body>
        <div class="object" id="object_1">                                  //---②
            object_1
            <div class="object" id="object_2">
                object_2
                <div class="object" id="object_3">
                    object_3
                    <div class="object" id="object_4">
                        object_4
                    </div>
                </div>
            </div>
        </div>
        <div class="button" id="button_1">Button 1</div>                    //---③
        <div class="button" id="button_2">Button 2</div>
        <div class="button" id="button_3">Button 3</div>
```

```
        <div class="button" id="button_4">Button 4</div>

        <script src="./js/object.js"></script>                    //---④
    </body>
</html>
```

① jQuery를 사용하기 위해서는 jQuery 파일을 사용해야 하는데, JavaScript 언어로 만들어진 jQuery 파일을 <script></ script> 태그로 가져다 사용하면 된다. 'jQuery' 파일의 이름을 'jquery.js'로 변경해서 사용해도 되지만, 'jquery-2.1.3.min.js'와 같이 jQuery 홈페이지(http://jquery.com)에서 다운받은 파일의 이름을 그대로 사용하는 이유는 현재 사용하고 있는 jQuery 파일의 버전을 아는 것이 편리하기 때문이다. 파일 이름에서 '2.1.3'은 파일의 버전Version을 의미하고 'min'은 파일의 크기를 줄이기 위해서 압축Encoding되었다는 것을 의미한다. JavaScript 파일들은 크기를 줄이기 위해서 압축을 많이 하는데, 다운받는데 시간이 절약될 뿐만 아니라 코드의 내용을 숨기기 위해서 상용 라이브러리Library들이 자주 사용하는 방법이다. 물론 무료로 배포되는 jQuery는 소스가 공개되었으며 오픈소스 프로젝트로 관리되기 때문에 GitHub 사이트(http://github.com/jquery)에서도 확인이 가능하다.

② 브라우저에 텍스트를 쓰기 위해서 네 개의 태그들이 사용되었으며, 상위 태그 안에서 하위 태그들이 선언되어 종속관계를 가지고 있다.

③ 네 개의 버튼들을 만들기 위해서 태그들이 사용되었으며, jQuery에서 버튼으로 만들어진 태그의 이벤트를 정의한다. 'jQuery'도 JavaScript 언어이기 때문에 [코드 1-34]의 'object.js'에서 구현되었다.

④ jQuery를 사용한 'object.js' 파일이 모든 태그들의 아래에 <script></script> 태그를 사용해서 정의된 이유는 태그들이 정의된 이후에 jQuery 함수들이 태크들의 객체에 접근해야 하기 때문이다. 이전 장에서 사용한 onload 이벤트를 사용해서 코드를 다 받은 후에 실행해도 되지만, 이와 같이 onload를 사용하지 않고 모든 태그들이 선언된 이후에 JavaScript 언어를 사용하는 방법을 많은 웹 프로그래머들이 즐겨 사용하고 있다. 만약 <body></body> 태그의 가장 아래에서 선언된 <script></script> 태그를 <head></head> 태그 안으로 옮기면 예제 코드는 정상적으로 동작하지 않는다.

[코드 1-34] jQuery 를 사용해서 태그 객체에 접근하고 다양한 동작을 수행하는 코드 (/1/object.js)

```
$(".button").hover(function() {                                   //---①
```

```
                $( this ).css({"background-color":"purple"});
            }, function() {
                $( this ).css({"background-color":"#1646a7"});
            });
$(".object").hover(function() {                                          //---②
                $( this ).css({"font-size":"40px"});
            }, function() {
                $( this ).css({"font-size":"25px"});
            });
$("#button_1").click(function() {                                        //---③
                $("#object_4").empty().html("Hello JisikSoft");          //---④
            });
$("#button_2").click(function() {                                        //---⑤
                $("#object_3").empty().html("Button_2 is Clicked.");
            });
$("#button_3").click(function() {                                        //---⑥
                $("#object_3").empty().html("object_3<div class='object'
                    id='object_4'>object_4 is added.</div>");
            });
$("#button_4").click(function() {                                        //---⑦
                $("#object_1").empty().html("for Beautiful World").css(
                    {"position":"relative",
                     "top":"80px",
                     "font-size":"60px"
                    }).hover(function() {
                                $( this ).css({"font-size":"70px",
                                    "color":"red"});
                            }, function() {
                                $( this ).css({"font-size":"60px",
                                    "color":"#1646a7"});
                    });
            });
```

① 'hover'는 "~의 위를 맴돌다."라는 의미가 있는데, 두 개의 이벤트Event onmouseover와 onmouseout을 합친 것으로 이해하면 된다. $(".button").hover()는 'button'이라는 클래스 값을 가진 모든 버튼에서 마우스가 버튼 위로 이동했을 경우와 버튼 밖으로 이동했을 때에 버튼의 색을 변경하는 것을 구현하였다. 'jQuery' 함수인 hover(function() { }, function() { })에서 JavaScript 함수를 의미하는 function()을 선언한 것은 hover() 함수 안에서 JavaScript 코드를 사용하기 위해서이다. 'hover()' 함수에서 먼저 선언된 "function() { }"에서는 마우스가 버튼 위로 이동했을 때에(onmouseover 이벤트) 수행해야 하는 코드를 구현했으며, 이후에 선언된 "function() { }"에서는 마우스가 버튼 밖으로 이동했을 때에(onmouseout 이벤트) 수행해야 하는 코드를 구현했다. 이 코드에서 중요한 것이 'this'의 사용인데, 'this'는 현재 자신이

선언된 곳에서의 객체를 의미한다. 즉, $(".button")으로 가져온 객체는 'button'이라는 클래스 값을 가지고 있는 모든 버튼 객체들이며, hover() 함수 안에서 사용된 this도 $(".button")으로 접근한 각각의 객체를 의미한다. JavaScript 언어를 사용해서 웹 프로그래밍을 할 때, this가 명시하는 객체를 이해하는 것은 상당히 중요하다.

② 화면의 글자들이 보이는 태그 객체에 마우스를 올리면 글자가 커지고, 마우스가 밖으로 나오면 원래 글자로 변한다. 여기서 중요한 것이 네 개의 태그 객체들은 종속성을 갖는데 하위 부분에 마우스를 옮기면 상위 태그 객체들의 글자도 커진다. 태그 객체들의 범위를 이해하면 된다. 즉, 하위 태그 객체에 마우스를 옮겨도 상위 객체의 범위에 포함된다.

③ jQuery의 click() 함수는 마우스를 클릭했을 때 수행되는 코드를 입력하는데, 여기서는 첫 번째 버튼('button_1')을 클릭했을 때 수행되는 동작을 구현한다.

④ 'object_4'라는 id 값을 가진 객체의 내용을 없애고 "Hello JisikSoft"라는 문자열 String을 넣는다. 여기서 중요한 것은 $("#object_4")는 태그 객체를 가져오고, empty() 함수를 수행한 이후에도 태그 객체로 존재하기 때문에 html() 함수를 다시 사용할 수 있다. 이와 같이, jQuery는 함수를 실행한 이후에도 객체로 존재하기 때문에 추가적인 함수들을 계속 사용할 수 있다.

⑤ 두 번째 버튼이 클릭되면 'object_3'이라는 id 값을 가진 태그 객체의 내용을 "Button_2 is Clicked." 문자열로 변경한다. 여기서 중요한 것은 'object_3' 값을 가진 태그 객체는 'object_4' 값을 가진 태그 객체를 포함하는데, 내용이 문자열로 변경되기 때문에 하위 객체인 'object_4' 값을 가진 태그 객체는 브라우저에서 사라진다.

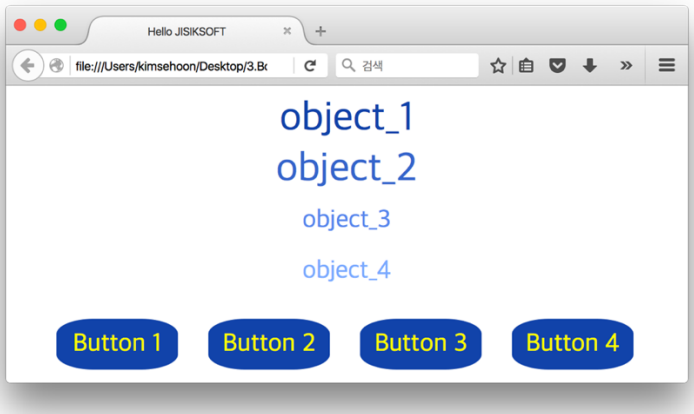
⑥ 두 번째 버튼과 마찬가지로 세 번째 버튼은 'object_3' 값을 가진 태그 객체의 내용을 변경하는데, 두 번째 버튼에서는 하위 태그 객체가 사라졌다면 세 번째 버튼에서는 'object_4' 값을 가진 하위 태그 객체를 만들어준다. 이와 같이 문자열을 사용해서 태그 객체들을 추가하는 방법을 이용해서 고급 웹 프로그래밍에서 동적인 Dynamic 프로그램의 구현을 가능하게 할 수 있다.

⑦ 네 번째 버튼이 선택되면 'object_1' 값을 가진 가장 상위 태그의 내용을 "for Beautiful World" 문자열로 변경하고 CSS 디자인을 적용한 후, hover() 함수를 사용해서 마우스가 옮겨졌을 때 글자의 크기가 커지고 붉은 색으로 변경되는 효과를 넣을 수 있다. 이와 같이 jQuery가 객체 중심으로 동작하는 것을 이해하면, jQuery의 함수들을 연속적으로 사용해서 다양한 프로그래밍 작업을 쉽게 할 수 있다.

[그림 1-56]부터 [그림 1-62]까지는 지금까지 구현한 코드의 결과를 보여주는데 프로그램을 실행해서 코드의 결과를 먼저 확인하는 것이 좋다. [그림 1-56]은 마우스가 글자 위로 이동하면 해당 글자와 위에 있는 상위 태그의 글자들이 함께 커지는 것을 알수 있다. [그림 1-56]은 마우스를 'object_2' 글자 위로 이동했을 때, 'object_1' 글자도 함께 커지는 결과를 보여준다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.4/1/object.html

그림 1-56 $(".object").hover() 함수에 의해서 마우스가 텍스트 위로 옮겨지면 크기가 변한다.



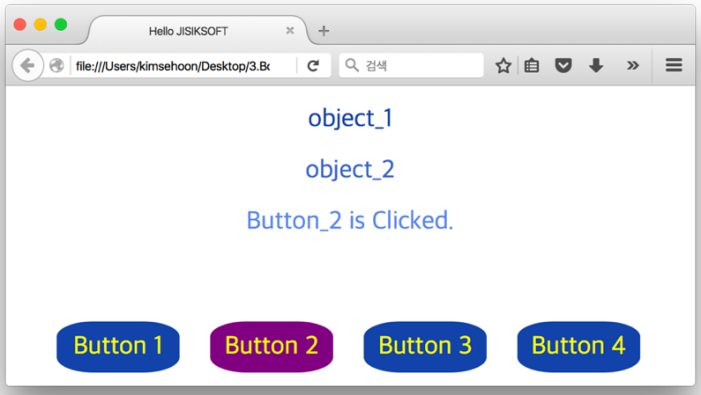
[그림 1-57]은 첫 번째 버튼("Button 1")이 선택되었을 때 'object_4' 값을 가진 마지막 태그의 내용이 "Hello JisikSoft"로 변경된 결과를 보여준다.

그림 1-57 $("#button_1").click() 함수에 의해서 마지막 텍스트가 "Hello JisikSoft"로 변경되었다.

[그림 1-58]은 두 번째 버튼("Button 2")이 선택되었을 때 세 번째 태그 객체의 내용을 "Button_2 is Clicked." 문자열로 변경하는데, 세 번째

태그 객체의 자식Child 태그인 네 번째 태그 객체가 사라졌다.

그림 1-58 $("#button_2").click() 함수에 의해서 "object_4" id 를 가진 태그가 사라졌다.



[그림 1-59]는 세 번째 버튼("Button 3")이 선택되었을 때, 세 번째 태그 객체에 사라졌던 네 번째 태그 객체를 넣어준 결과를 보여준다. 여기서 중요한 것이 새로 추가된 태그에는 코드에서 사용한 $(".object").hover() 함수가 동작하지 않는데, [그림 1-60]과 같이 마우스를 "object_4 is added." 문자에 올려도 해당 글자가 커지지 않는다. 코드에서 사용한 jQuery의 hover() 함수는 프로그램이 시작되면 한 번만 실행되기 때문에 새롭게 만들어진 객체는 다른 객체들과 같은 클래스 값을 가졌다 하더라도 hover() 함수의 동작이 적용되지 않는다. 새롭게 만들어진 객체에 jQuery 함수를 다시 실행해서 객체에 효과를 주어야 하는 것을 숙지할 필요가 있는데, JavaScript에서는 디자인 효과를 준다는 것은 해당 객체마다 변경된 효과들을 개별적으로 저장하고 있다는 개념을 이해하게 된다.

그림 1-59 $("#button_3").click() 함수에 의해서 "object_3" 태그 안에 "object_4" 태그가 추가되었다.

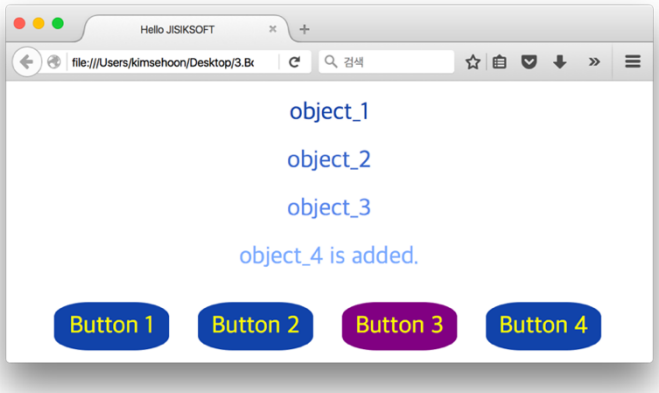


그림 1-60 "Button 3"의 동작 후에 생성된 "object_4"에서는 텍스트 크기가 변하지 않는다.

[그림 1-61]은 네 번째 버튼("Button 4")이 선택되었을 때, 'object_1'의 id 값을 가진 가장 상위의 태그의 내용을 "for Beautiful World"로 변경했으며, [그림 1-62]는 디자인 속성이 변경되어서 마우스가 글자위로 이동하면 글자가 커지고 붉은 색으로 변하는 것을 보여준다. 여기서 중요한 것은 네 번째 버튼이 선택된 이후에는 왼쪽에 있는 세 개의 버튼이 동작하지 않는다는 것을 이해해야 한다. 즉, 가장 상위의 태그의 내용이 변경되면서 종속된 자식 태그들이 모두 사라졌기 때문에, 왼쪽의 버튼을 클릭해도 접근할 수 있는 태그 객체가 존재하지 않기 때문에 어떠한 동작도 수행되지 않는다. 고급 웹 프로그래밍은 태그 객체를 프로그램이 실행되는 중간에 만들고 지울 수 있다는 것을 이해하는 것이 중요하다.

그림 1-61 $("#button_4").click() 함수에 의해서 "object_1" 태그만 남고 나머지 태그는 없어졌다.

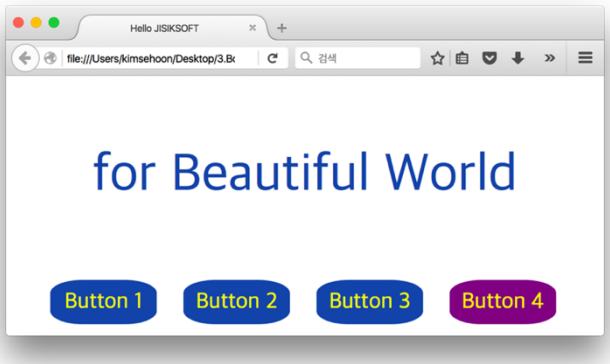
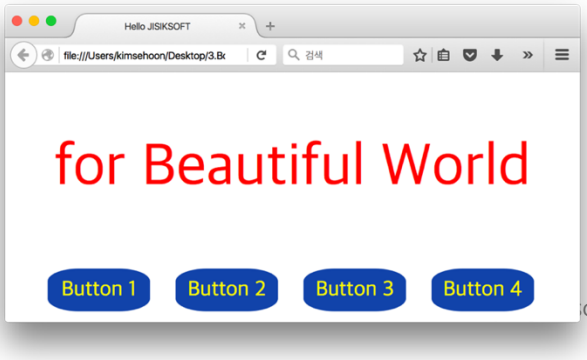


그림 1-62 "Button 4" 클릭 후 "object_1" 태그만 존재하기 때문에 다른 버튼들은 동작하지 않는다.



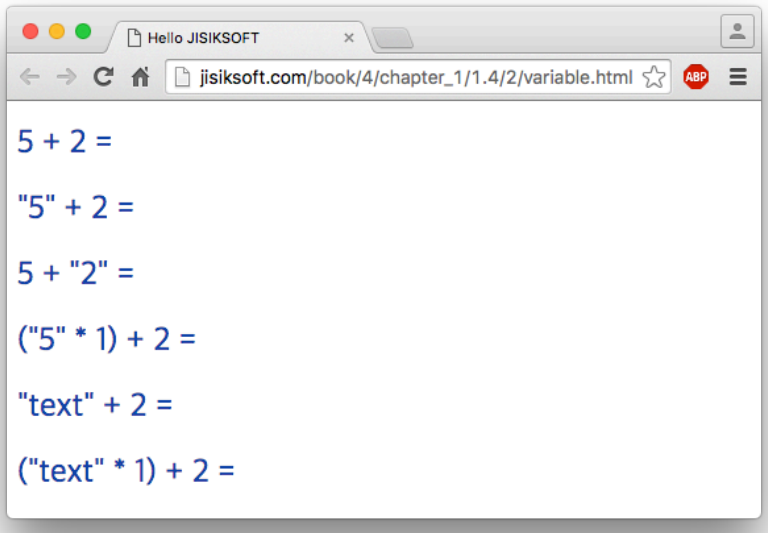
JavaScript 언어의 사용이 쉽다고 여겨지는 이유 중의 하나는 변수Variable의 데이터형Data Type을 구분해서 사용할 필요가 없다는 것인데, 프로그래머의 입장에서는 다소 편리하게 여겨질 수 있지만 변수의 값이 어떻게 변하는지를 정확히 이해하고 사용하는 것이 필요하다. 변수를 선언하는 'var' 예약어를 사용해서 모든 문자열String과 숫자Number 값을 갖는 변수를 선언할 수 있지만, 문자열로 선언된 변수가 숫자 값으로 변하고 숫자로 선언된 값이 문자열로 변하는 것을 이해해야 JavaScript의 변수를 정확히 사용할 수 있다. 연산자 중에서 '+'(더하기) 연산자는 숫자들의 연산에서는 더하기를 수행하고, 문자열에서는 두 개의 문자열을 연결하는 기능을 한다. 또한, 변수의 내용이 숫자로 이루어졌어도 변수는 숫자로 기억할 수도 있고 문자열로 인식할 수도 있는데, 이러한 이유 때문에 숫자로 이루어진 문자열을 정수나 실수로 변경하고 수Number로 이루어진 데이터를 문자열로 쉽게 변경할 수 있다.

[그림 1-63]은 문자열과 숫자의 형 변환을 이해하기 위한 코드의 시작 화면을 보여주는데, 코드의 구현 후에 알게되는 결과를 보기 전에 독자분들이 재미있는 퀴즈로 생각하고 풀어보기를 권한다.

[코드 1-35]와 [코드 1-36]은 jQuery의 click() 함수를 사용해서 연산을 수행하는 동작을 구현했다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.4/2/variable.html

그림 1-63 변수의 특성을 확인하기 위한 코드의 시작 화면



[코드 1-35] 문자열과 숫자의 연산에서 모두 사용하는 '+' 연산자를 비교하는 코드 (/2/variable.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./css/variable.css"></link>
        <script src="./js/jquery-2.1.3.min.js"></script>
    </head>

    <body>
        <div class="variable" id="variable_1">5 + 2 = </div>                    //---①
        <div class="variable" id="variable_2">"5" + 2 = </div>
        <div class="variable" id="variable_3">5 + "2" = </div>
        <div class="variable" id="variable_4">("5" * 1) + 2 = </div>
        <div class="variable" id="variable_5">"text" + 2 = </div>
        <div class="variable" id="variable_6">("text" * 1) + 2 = </div>

        <script src="./js/variable.js"></script>                                //---②

    </body>
</html>
```

① 여섯 개의 태그들을 사용해서 연산을 수행하는 문자열을 브라우저에 그려준다.

② 태그들이 클릭되었을 때, 연산의 결과를 오른쪽에 보여주는 코드들을 구현했다. 태그들이 브라우저에 그려지고 난 이후에 수행되는 JavaScript 코드이기 때문에 태그의 제일 아래에 <script></script> 태그를 사용해서 추가되었다.

[코드 1-36] 문자열과 숫자의 연산을 수행하는 JavaScript 코드 (/2/variable.js)

```
$("#variable_1").click(function() {                                             //---①
        var value = 5;
        value = value + 2;
        $(this).append(value);                                                  //---②
 });
$("#variable_2").click(function() {
        var value = "5";
        value = value + 2;
        $(this).append(value);
 });
$("#variable_3").click(function() {
        var value = 5;
```

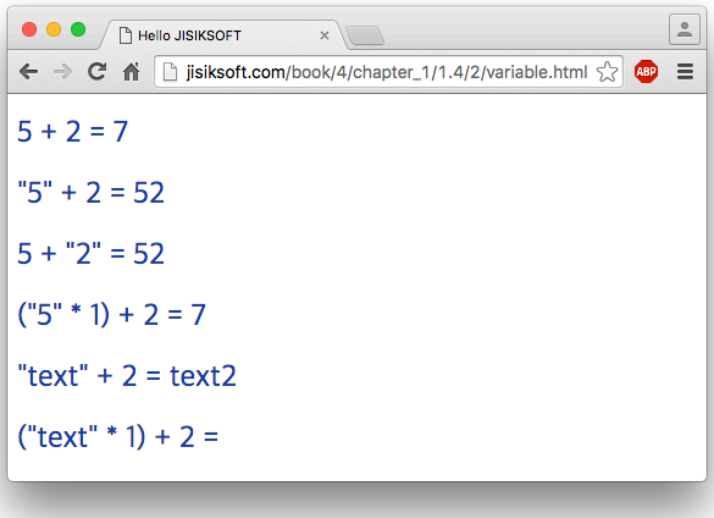
```
        value = value + "2";
        $(this).append(value);
 });
$("#variable_4").click(function() {
        var value = "5";
        value = (value * 1) + 2;
        $(this).append(value);
 });
$("#variable_5").click(function() {
        var value = "text";
        value = value + 2;
        $(this).append(value);
 });
$("#variable_6").click(function() {
        var value = "text";
        value = (value * 1) + 2;
        $(this).append(value);
 });
```

① 태그들이 클릭되었을 때 수행되는 JavaScript 코드들을 jQuery의 click() 함수를 사용해서 구현하였다.

② this는 jQuery의 click() 함수가 실행할 때의 객체를 가리키기 때문에, 여기서는 선택된 태그 객체이며 append() 함수를 사용해서 객체의 마지막에 연산의 결과를 더해준다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.4/2/variable.html

그림 1-64 브라우저의 연산을 모두 클릭한 이후의 결과

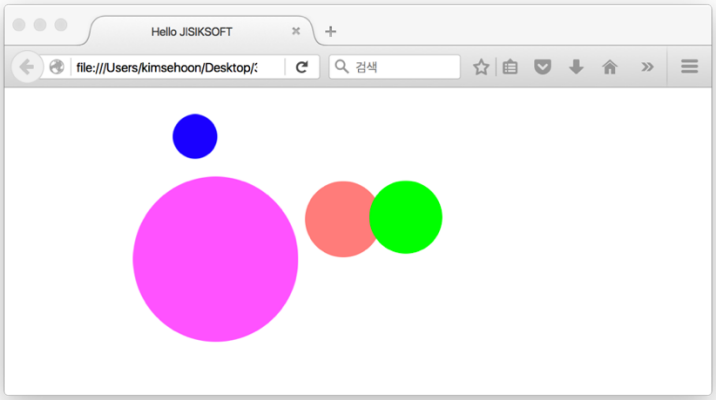


'jQuery'는 JavaScript로 구현할 수 있는 많은 기능들을 간단한 함수로서 쉽게 사용

할 수 있게 만들었는데 그 중에서 가장 대표적인 것이 animate() 함수다. 'animate()' 함수는 설정된 시간(milli-second)동안 태그의 디장인 속성에 변화를 주게 되는데, 디자인 속성의 값들이 단계적으로 변하기 때문에 움직이는 효과를 줄 때 사용한다. 주로 사용하는 방법이 태그 객체의 위치를 변경할 때 사용하는데, 시작되는 위치에서 설정된 위치로 값이 서서히 변하면서 브라우저에 그려주기 때문에 사용자는 해당 태그가 움직이는 동작을 보게 된다. 이번에 구현하는 프로그램은 네 개의 원이 크기와 위치가 변하면서 브라우저에서 움직이는 프로그램을 구현하는데, [그림 1-65]는 브라우저에 네 개의 원이 존재하는 것을 보여준다. 아래의 "확인 사이트"에서 프로그래밍 어떻게 동작하는지를 미리 보고 [코드 1-37]부터 [코드 1-39]까지의 코드를 보는 것이 좋다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.4/3/animate.html

그림 1-65 jQuery 의 animate() 함수로 동적인 화면을 만들 수 있다.



[코드 1-37] 브라우저에서 움직이는 원들을 만드는 HTML 코드 (/3/animate.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./css/animate.css"></link>
        <script src="./js/jquery-2.1.3.min.js"></script>
        <script src="./js/animate.js"></script>
    </head>

    <body onload="initialize();">                                    //---①
```

```
        <div class="circle" id="circle_1"></div>                    //---②
        <div class="circle" id="circle_2"></div>
        <div class="circle" id="circle_3"></div>
        <div class="circle" id="circle_4"></div>
    </body>
</html>
```

① onload 이벤트를 사용해서 브라우저에서 모든 코드를 받은 후에 네 개의 사각형을 움직이는 'initialize()' 함수를 실행한다.

② 네 개의 태그를 사용해서 네 개의 원을 그리는데, 원은 가로와 세로가 동일하며 테두리를 원으로 그려주면 되는데 [코드 1-38]의 CSS 디자인에서 설정했다.

[코드 1-38] 4 개 원들의 디자인을 위한 CSS 코드 (/3/animate.css)

```
.circle {
    position:absolute;
    border-radius:50%;                                              //---①
    width:50px;
    height:50px;
}
#circle_1 {
    background-color:#ff0000;
}
#circle_2 {
    background-color:#00ff00;
}
#circle_3 {
    background-color:#0000ff;
}
#circle_4 {
    background-color:#ff00ff;
}
```

① 원을 만들기 위해서는 태그의 테두리를 동그랗게 만들면 되는데, 가로Width와 세로Height의 크기를 동일하게 한뒤에 'border-radius' 속성을 50%로 설정하면 된다.

[코드 1-39] 브라우저에서 4 개의 원을 움직이는 Javascript 코드 (/3/animate.js)

```
var width, height;                                                  //---①

move = function(container) {                                        //---②
```

```
    var opacColor = Math.random() + 0.3;                                    //---③
    var posX = Math.random() * (width-10);                                  //---④
    var posY = Math.random() * (height-10);
    var moveTime = Math.random() * 2000 + 500;                              //---⑤
    var delayTime = Math.random() * 5000;                                   //---⑥
    var sizeCircle = (Math.random() * 150) + 20;                            //---⑦

    var distWidth = 0;                                                      //---⑧
    if ((posX + sizeCircle) > width)
        distWidth = width - sizeCircle;
    var distHeight = 0;
    if ((posY + sizeCircle) > height)
        distHeight = height - sizeCircle;

    $( "#"+container ).animate({                                            //---⑨
                                    opacity: opacColor,
                                    left: (posX - distWidth)+'px',
                                    top: (posY - distHeight)+'px',
                                    width: sizeCircle+'px',
                                    height: sizeCircle+'px'
                                }, moveTime);

    setTimeout(function(){ move(container); }, delayTime);                  //---⑩
}

initialize = function() {                                                   //---⑪

    width = $(window).width();                                              //---⑫
    height = $(window).height();

    move("circle_1");                                                       //---⑬
    move("circle_2");
    move("circle_3");
    move("circle_4");
}
```

① 브라우저의 가로Width와 세로Height 길이를 저장하기 위한 변수로 width와 height가 사용되었다.

② 하나의 원을 크기와 위치를 변경해서 서서히 움직이는 동작을 수행하며, 원의 id 값을 매개변수인 container 값으로 받는다.

③ 투명도Opacity를 변경하기 위해서 사용되는 값을 저장하기 위해서 opacColor 변수를 사용했으며, 임의의 값Random Value을 생성하기 위해서 Math.random() 함수

를 사용했다. Math.random() 함수는 0과 1 사이의 임의의 실수를 반환하는데, 0이면 원의 뒷 배경이 모두 나오고 1이면 투명도가 없다는 의미이다. 여기서는 투명도가 0.3 이하로는 내려가지 않게 하기 위해서 0.3을 더해주었다.

④ 원의 위치값을 저장하기 위해서 posX와 posY 변수들을 사용하는데, 임의의 값을 갖기 위해서 Math.random() 함수를 사용하고 여기에 브라우저의 가로나 세로의 길이를 곱해주었다.

⑤ moveTime 변수는 원이 움직이는 시간(milli-second) 값을 갖는데 0과 1사이의 임의의 실수에 2000(2초)을 곱해주었으며, 0.5초(500ms) 이하로 빠르게 움직이는 것을 방지하기 위해서 500을 더해주었다.

⑥ 하나의 원은 움직이고 멈추고를 반복하는데, 이동한 이후에 멈춰있는 시간을 갖기 위해서 0과 1사이의 임의의 실수값에 5000(5초)을 곱해주었다. 원은 움직이고 난 이후에 5초 미만의 시간 동안 잠시 멈추게 하기 위해서 delayTime 변수를 사용했다.

⑦ 원을 그리기 위한 사각형의 태그는 가로Width와 세로Height의 값이 같으며, 가로와 세로의 값을 위하여 sizeCircle 변수를 사용했다. 크기는 20보다 작지 않으며 170(=150+20)보다 크지 않은 값을 갖는다.

⑧ 사각형이 브라우저 밖으로 나가지 않아야 하기 때문에, distWidth 값은 오른쪽 밖으로 나가는 길이 값이 계산되어 저장되며 distHeight는 아래쪽 밖으로 나가는 길이 값을 갖는다.

⑨ jQuery의 대표적인 함수인 animate()를 사용해서 원의 투명도, 위치, 크기를 moveTime 값에서 정해진 시간동안 서서히 변화를 준다. $("#"+container)는 원의 태그 객체 값을 의미하며 animate() 함수는 "animate({ CSS 변경 내용 }, 변화시간);"와 같이 두 개의 매개변수를 갖는다.

⑩ 원이 움직인 다음에는 잠시 멈추게 되는데, setTimeout() 함수를 사용해서 delayTime 시간 이후에 움직이는 동작을 진행하는 move() 함수를 다시 실행했다.

⑪ 브라우저에서 모든 파일을 받은 후에 onload 이벤트를 통해서 실행되는 함수가 initialize() 함수이며, 브라우저의 크기를 계산하고 네 개의 원을 차례대로 움직이게 한다.

⑫ 브라우저의 가로와 세로의 크기를 얻기 위해서는 브라우저의 화면Window 객체를 가져와야 하는데 jQuery에서는 $(window)를 사용해서 화면 객체를 가져올 수 있다. 다른 모든 객체에서 사용할 수 있는 width()와 height() 함수들을 사용해서 브라우저의 크기를 알수 있다.

⑬ 네 개의 객체들의 id 값을 가지고 move() 함수들을 실행시키면 네 개의 원이 브라우저 안에서 움직이는 효과를 볼 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.4/3/animate.html

그림 1-66 jQuery 의 animate() 함수로 브라우저에서 4 개의 동그라미들이 주기적으로 움직인다.

jQuery를 잘 사용하기 위해서는 웹 프로그래밍에서 객체Object의 개념을 정확히 이해해야 하며, jQuery 함수들을 사용해서 객체에 다양한 변화를 주는 것이 가능하다. 많은 사람들이 jQuery를 잘해야 웹 프로그래밍을 잘 하는 것이라 생각하는데, 아주 많은 함수들을 가지고 있는 jQuery의 모든 것을 알기보다는 이번 장에서 설명한 내용들을 깊이 있게 이해한 후에 필요한 jQuery 함수를 찾아서 사용하는 것이 좋다. 필자의 경험으로는 animate() 함수만을 사용해서도 많은 다양한 웹 프로그램들을 구현할 수 있었다. 프로그래머는 많은 것을 기억하는 것이 중요한 것이 아니라, 해당 언어의 특징을 잘 알고 무엇이 프로그래밍에서 필요한지를 알고 잘 찾아서 사용하는 능력이 더 중요하다고 필자는 생각한다. 그러한 과정이 반복되면서 프로그래밍 언어에서 자유로와지며, 다른 새로운 언어를 배우더라도 쉽고 빠르게 해당 언어를 익힐 수 있다.

1.5 PHP

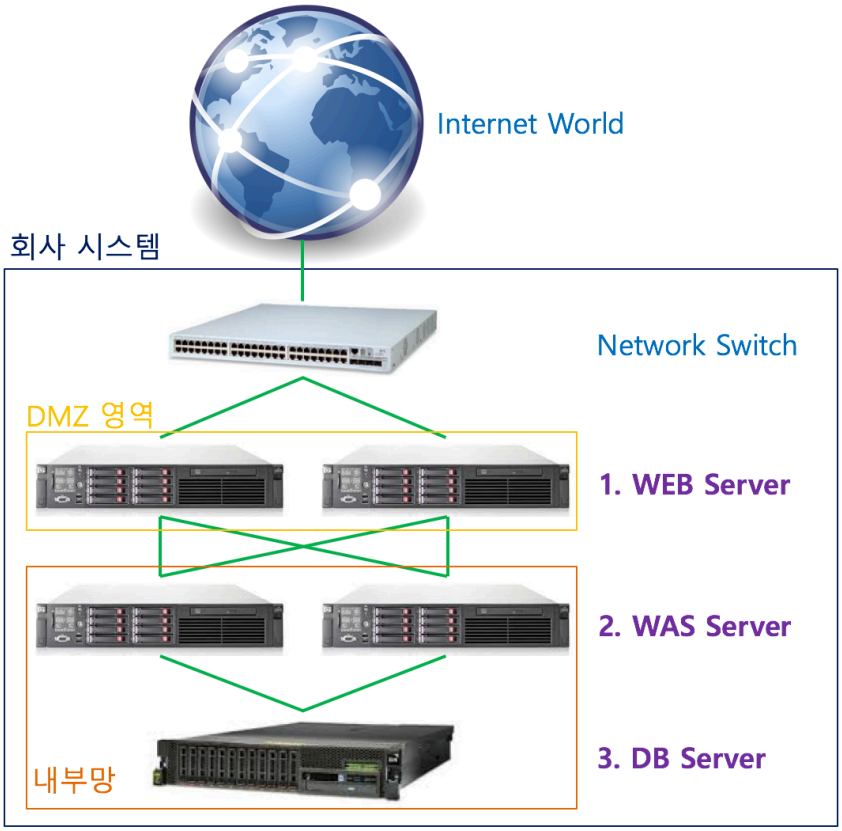
이전 장까지는 브라우저에서 동작하는 내용들을 다루었고, 이번 장에서 설명하는 PHP는 서버에서 동작하는 언어다. 서버를 운영할 필요가 없는 일반 사람들은 인터넷의 정보만을 이용하면 되지만, 웹 프로그래머는 서버의 기본적인 내용들을 알아야

하기 때문에 서버 시스템에 관해서 먼저 설명하고자 한다. 기본적으로 가장 많이 사용하고 있는 서버 시스템이 Linux 서버인데, 소규모 사업장에서는 비용적인 측면으로 인해서 클라우드Cloud 서비스를 제공하는 회사에 월 사용료를 지불하고 Linux 서버를 사용하는 경우가 많다.(일반적으로 대부분의 Linux는 무료로 배포된다.) 대부분의 서버가 일반 PC보다 성능이 좋은데, 일반 가정집에 놓고 사용하기에는 소음이 너무 크다. 물론, 일반 PC에 Linux를 설치하고 서버로 사용할 수도 있지만, 백업Backup이 이루어지지 않는 PC에 데이터를 저장하는 것은 상당히 위험하고, 전기세와 장소 등을 고려하면 클라우드 서버를 사용하는 것이 올바른 선택일 수 있다. 웹 프로그래밍을 하는 사람이라면 월 몇만 원의 비용을 클라우드 회사에 지불하고 개인 서버를 운영하는 것을 권하며, 필자가 몸담고 있는 '지식소프트' 회사도 클라우드 서버 몇 대를 운영하고 있다. '지식소프트' 같은 작은 회사에서는 많은 사용자가 방문하지도 않고 중요한 데이터가 거의 없기 때문에 1대의 서버로 모든 웹 서비스를 수행하지만, 큰 회사의 시스템에서는 다수의 서버를 기능에 따라 분리해서 사용한다.

[그림 1-67]은 규모가 큰 회사의 시스템 구성도를 간단히 보여주는데, 서버의 역할에 따라서 WEB Server(웹 서버), WAS Server(와스 서버), DB Server(디비 서버)로 구성된다. 일반적으로 DB Server는 모든 정보를 한곳에 저장하기 때문에 한 대를 운영하며 가장 성능이 좋은 하드웨어Hardware와 가장 안정적인 운영체제인 Unix를 사용한다. WEB Server와 WAS Server들은 사용자가 많은 회사일 경우에는 다수의 Linux 서버들로 구성하는데, WEB Server는 회사의 입장에서 바라보면 외부라고 할 수 있는 인터넷상에서 접근이 가능한 DMZ 영역에 설치하고, WAS Server는 외부에서 절대 접근할 수 없는 내부망에 DB Server와 같이 설치한다. 시스템 담당 부서에서 일하게 되면 서버를 설치할 때 내부망에 설치해야 하는지, 아니면 DMZ 영역에 설치하는지가 상당히 중요한데 회사의 내부망이 보안에 취약해서 공격을 받게 되면 심각한 문제가 발생하기 때문이다. 남북한으로 나누어진 우리나라는 휴전선의 비무장 지대를 DMZ라고 하기 대문에 DMZ라는 용어를 많이 접하는데, 회사의 시스템 네트워크 상에서도 외부망(Internet)과 내부망의 완충지대 역할을 하는 영역을 만들어서 DMZ라고 부르고 있다. 즉, 내부망과 외부망이 선으로 직접 연결되면 시스템을 공격하는 해커Black Hacker들이 수시로 공격을 시도해서 데이터베이스의 정보를 탈취하려고 할 수 있기 때문에 DMZ 영역을 만들어서 외부와의 직접적인 연결을 차단한다. 예를 들어, 온라인 은행 업무에서 일반 고객이 접근하는 것은 WEB Server이며, 이체를 한다는 것은 WEB Server를 통해서 WAS Server에 접근하고 DB Server의 데이터 변경

이 이루어지는 일련의 행위라고 보면 된다. 오래 전 우리나라의 한 은행에서 불법적인 행위가 있었는데 내부망에 연결된 개인 노트북을 통해서 악성 프로그램이 내부망의 WAS Server로 감염되어 발생한 것으로 이해하면 된다. 그래서 은행처럼 보안이 중요한 시스템에서는 외부로부터의 컴퓨터 반입을 엄격히 통제하고 있다. WEB Server는 일반적으로 브라우저에 보여지는 코드와 이미지들을 저장하고 있으며, 인터넷에서 사용자의 요청에 따라서 해당 데이터들을 전송한다. WAS Server는 '로그인'과 같이 DB Server의 정보가 필요할 때 주로 사용되는데, 사용자의 요청에 대하여 단순히 정보를 전달하는 것이 아닌 프로그래밍적인 방법을 수행하는 서버로 이해하면 된다. DB Server는 관리해야 하는 모든 데이터들을 저장하고 있는 서버이며, 비밀번호와 같은 중요한 데이터들이 많기 때문에 회사의 가장 핵심적인 서버로 이해하면 된다. [그림 1-67]에서 스위치라는 장비는 네트워크 상에서 신호등과 같은 역할을 하는데 무수히 많은 전설들이 있는 서버실에서 서버들간의 연결을 만들어준다. 그림에서는 인터넷과 직접적으로 연결된 것은 가장 앞에 있는 스위치 장비이며 사용자의 요청에 따라서 WEB Server로 데이터를 분산시키는 역할을 한다. 물론, 서버들 간에 직접적인 연결이 이루어지는 것이 아니라 내부망에서도 다수의 스위치 장비들이 설치되지만 [그림 1-67]에서는 시스템 구성도를 단순화 시키기 위해서 생략했다.

그림 1-67 다수의 서버들을 사용하는 회사의 시스템 구성도

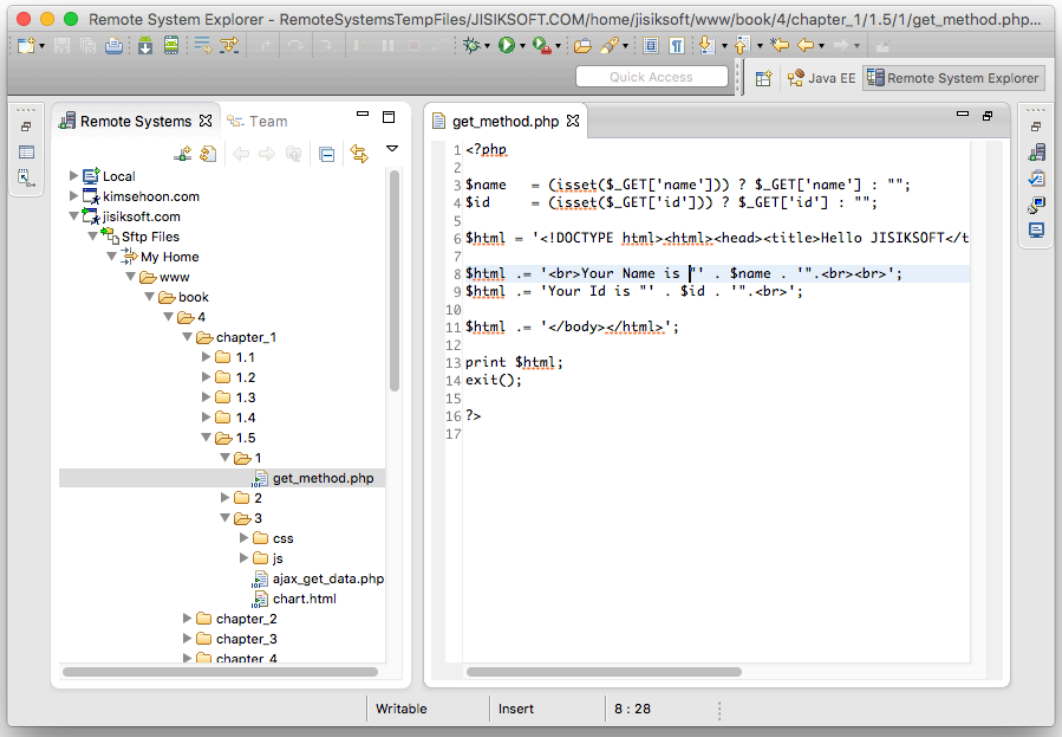


[그림 1-67]에서는 규모가 큰 회사의 시스템 구성도를 보여준다. '지식소프트'와 같은 소규모 회사는 한 대의 WEB Server만을 사용하며 WEB Server에 DB[DataBase]를 설치해서 모든 웹 서비스를 하도록 만들었다. 보안에 취약하더라도 회사의 운명을 좌우하는 데이터가 서버에 없기 때문에 가능하며 비용적인 측면을 고려하면 클라우드[Cloud] 서비스로 한 달에 몇 만원으로 서버를 운영할 수 있다. 이제부터 WEB Server를 웹 서버라고 부르고, 웹 서버 한대에서 모든 웹 서비스를 제공하는 것으로 가정하고 이해하길 바란다. 여기서 사용하는 웹 서버는 Linux 운영체제를 사용하고, 웹 서버를 만들기 위해서 Apache라는 프로그램을 설치했다. Apache는 외부의 요청이 있을 때, 정해놓은 위치의 파일 내용을 HTTP 규약[Protocol]에 기반해서 보내주는 역할을 한다. 프로그래머의 관점에서 보면, Apache는 'index.html'이라는 파일을 요청 받으면 'index.html' 파일을 열어서 서버 메모리[Memory]에 올리고 HTTP 규약에 따라서 필요한 정보를 먼저 보내고 'index.html' 파일의 내용을 보내주는 역할을 하는 프로그램이라고 이해하면 된다. 사용자 관점에서는 Apache가 설치된 웹 서버로부터 정보를 받은 브라우저가 'index.html'에 있는 내용대로 보여주는 역할을 하는 것이다. 이전 장에서 설명한 브라우저에서만 동작하는 코드들만 가지고도 웹 서버는 운영될 수 있는데, WWW(World Wide Web)이 처음 사용된 1990년대 말에는 이렇게만 만들어진 인터넷 세상이었다. 그러나, 정보만을 보여주는 것이 아니라 사용자의 요구에 따른 다양한 기능을 하는 지금의 인터넷 세상으로 발전하면서 웹 서버에서도 처리해야할 것들이 많아졌으며, 웹 서버에서 동작하는 프로그래밍 언어들이 필요하게 되었다. 그래서 우리가 사용하는 것이 JSP, PHP, ASP와 같은 언어들이며, JSP는 Java 계열이고, PHP는 C/C++ 사용자에게 좀더 친숙하고, ASP는 Windows 운영체제에서 동작하는 언어로 이해하면 된다. 현재는 Java 프로그래머가 가장 많으며, 많은 대형 시스템에서 사용하는 언어는 JSP이다. 그러나, '지식소프트'와 같은 작은 회사에서는 PHP를 많이 사용하고 있으며 이 책에서도 PHP를 사용해서 웹 프로그래밍을 구현한다. 웹 서버에서 사용하는 언어를 정하는 것은 Apache를 설치해서 웹 서버를 만들고, 사용하려는 언어를 설치하면 된다. Apache와 PHP를 설치한 Linux 서버를 기반으로 이 책이 만들어진 것이며, Apache와 PHP 설치와 사용법에 대해서는 인터넷에 많은 자료가 있기 때문에 여기서는 생략한다. 이 책은 개념적인 부분을 중심으로 최대한 간략히 설명하고자 하기 때문에 이 책에서 설명하지 않은 많은 내용들은 인터넷에서 정보를 찾아서 구현해 나가기를 필자는 바란다.

이번 장부터는 웹 서버에서 동작하는 PHP 프로그래밍을 하기 때문에 독자분들도

무료로 제공되는 클라우드Cloud 서버나 1년에 만 원 미만으로 사용할 수 있는 호스팅 서비스를 이용해서 웹 프로그래밍을 계속하기를 바란다. 개인적으로 Linux 서버를 사용하지 않고 Linux 서버에서 홈페이지 서비스를 제공하는 일반적인 호스팅 업체를 이용하는 방법이 있는데, 이러한 경우에는 호스팅 업체에서 제공하는 계정의 환경에는 Apache와 PHP가 이미 설치된 경우가 많다. 호스팅 계정으로 접속을 하게 되면 'www/' 폴더Directory가 만들어져 있는 경우도 있는데, 예를 들어 브라우저에서 지식소프트 홈페이지(http://jisiksoft.com)에 접속하면 Apache는 'www/' 폴더에서 'index.html' 파일을 읽어서 전송하는 역할을 수행한다. 호스팅 서비스를 사용하는 독자들은 'www/' 폴더 안에 필요한 파일이나 폴더를 만들어서 사용하면 된다. 서버에 있는 파일들을 가지고 프로그래밍을 하기 위해서는 원격Remote으로 접속해서 파일을 주고 받거나 프로그래밍을 할 수 있는 에디터가 필요한데, 가장 많이 사용하는 무료 소프트웨어는 'FileZilla' 파일전송 프로그램과 'Eclipse' 에디터Editor이다. 필자는 현재 'FileZilla'와 'Eclipse' 두 개의 소프트웨어를 주로 사용하는데, 이와 같은 기능을 하는 소프트웨어들은 많기 때문에 독자들도 자신에게 맞는 소프트웨어를 사용하면 된다.

그림 1-68 원격으로 파일을 편집하고나 관리할 수 있는 에디터 프로그램인 Eclipse

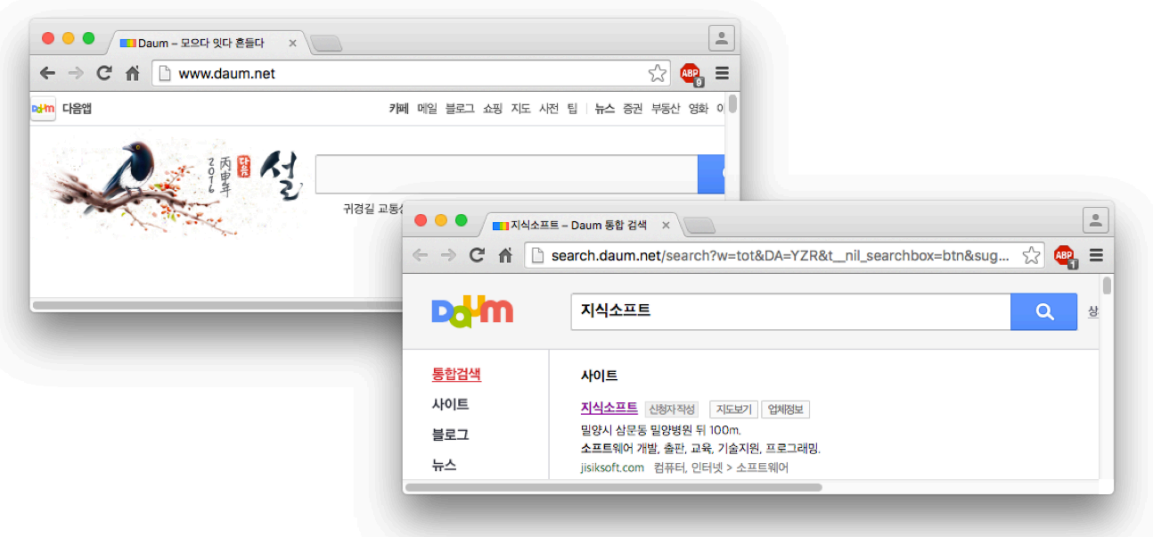


[그림 1-68]은 Eclipse 에디터의 이미지를 보여준다. 왼쪽에는 접속한 웹 서버의

'www/' 폴더의 파일 구조들을 보여주며 오른쪽에서는 선택된 파일들을 편집할 수 있는 환경이 제공된다. Eclipse는 Java 개발자들이 가장 많이 사용하는 에디터이며, Java에 기반한 안드로이드 모바일 프로그래밍 환경을 잘 제공해주고 있기 때문에 인기가 많은 편이다. 프로그래밍 언어를 선택하는 것과 프로그래밍 에디터를 정하는 것은 전혀 별개의 문제이기 때문에 자주 사용하는 에디터가 본인에게 가장 좋은 환경을 제공할 수 있다.

브라우저에서 인터넷을 통해 서버에 요청할 때는 HTML 페이지만을 가져올 수도 있지만, 서버에 특정 정보를 담아서 요청하는 경우도 많다. 예를 들면, 검색 사이트인 Daum 홈페이지(http://daum.net)를 열 때는 주소 창에 "http://daum.net"이라고만 입력하면 첫 페이지가 열린다. 하지만 Daum 검색창에서 '지식소프트'라고 입력하면 주소 창에 많은 내용들이 보여지는데, [그림 1-69]는 Daum 사이트에서 '지식소프트'를 검색한 결과를 보여준다.

그림 1-69 Daum 검색 사이트에서 '지식소프트'를 검색한 결과



Daum 사이트에서 '지식소프트'를 검색했을 때, 주소 창에 보여지는 내용은 아래와 같으며 마지막에 '지식소프트'라는 단어가 포함된 것을 볼 수 있다.

"http://search.daum.net/search?w=tot&DA=YZR&t__nil_searchbox=btn&sug=&sugo=&sq=&o=&q=지식소프트"

위와 같이 브라우저의 주소 창에 데이터를 넣어서 서버에 요청하는 방법을 웹 프로그래밍에서는 "Get 방식"이라고 한다. "Get 방식"과 비교되는 방법으로 "Post 방식"이 있는데, 브라우저의 주소 창에 서버로 보내는 데이터가 보이지 않고 전달되는 방

법이다. 사용자가 서버로 전달하는 데이터가 브라우저 주소 창에 보이느냐의 차이만 있을 뿐 네트워크로 전송되는 데이터를 확인하는 프로그램을 사용하면 확인은 가능하다. 'Get 방식'은 'Post 방식'보다 사용이 편리할 수 있지만 주소 창에 불필요한 데이터 값들이 보이는 것을 사용자들은 좋아하지 않는다. HTML은 주고 받는 데이터를 중간에서 가로채는 공격자가 있으면 숨길 수 없는 약점이 있는데, 이것을 보완한 방법이 데이터를 암호화해서 주고 받으면 해결되며, 인터넷에서 이것을 가능하게 한 것이 HTTPS(Secure HTTP: 원래는 다른 이름이지만 이렇게 이해하는 것이 쉽다.) 규약이다. HTTPS는 암호화를 인증하는 기관에서 발행한 인증서를 받아서 서버와 브라우저에서 암호화를 해서 데이터를 주고받기 때문에 중간에서 데이터를 가로채더라도 확인이 불가능하다.

[그림 1-70]은 'Get 방식'이 주소 창에서 사용될 때의 구조를 보여주는데, 세 개의 특수문자들('?', '&', '=')을 사용해서 브라우저 주소 창에 데이터들을 입력한다. 즉, 서버로 전달하는 데이터들은 '?' 문자 뒤에 써주며, 데이터들간의 구분은 '&'를 사용한다. 'Get 방식'은 데이터를 요청하는 것이 아니고, 서버의 프로그램을 실행하는 것이기 때문에 PHP 언어로 만들어진 .php 확장자를 가진 실행 파일을 주소 창에서 사용한다. PHP 언어는 컴파일을 해서 실행파일을 만들어서 사용하는 언어가 아니라 코드의 내용을 해석해서 실행하는 JavaScript와 같은 스크립트Script 언어이기 때문에 C/C++와 전혀 별개의 언어이다. Java를 사용하는 JSP는 컴파일을 한 후 사용하기 때문에 속도가 PHP보다 상대적으로 훨씬 빨라서 대형 시스템에서는 JSP로 서버 시스템을 구축하는 것이라 이해하면 된다. PHP는 사용자가 많지 않은 '지식소프트'와 같은 소규모 회사에서 개발하기가 수월하다는 장점이 있다.

그림 1-70 서버로 데이터를 전달하는 Get 방식 구조

<Get 방식 구조>

http://jisiksoft.com/get_method.php**?**name=JisikSoft**&**id=Hello

- '**?**' : 실행파일과 전달되는 데이터를 구분한다.
- '**&**' : Get 방식으로 전달되는 데이터들을 구분한다.

사이트 주소 : http://jisiksoft.com
실행 파일 : get_method.php
Get 방식 데이터 : name**=**JisikSoft (변수: name, 값: JisikSoft)
id**=**Hello (변수: id, 값: Hello)

- '**=**' : Get 방식으로 전달되는 변수와 값을 구분한다.

[코드 1-40]은 'Get 방식'으로 받은 데이터를 처리하는 PHP 코드를 보여주며, 전달받은 데이터의 변수 이름으로 값을 받아서 서버에서 프로그램이 실행되고 결과를 사용자의 브라우저로 보내주는 구조를 갖는다. 아래의 '확인 사이트'를 통해서 'get_method.php' 파일을 실행한 후에 받는 결과를 확인할 수 있으며, 주소 창에서 전달하는 데이터의 'name'과 'id'의 값을 변경하면 브라우저의 결과가 새롭게 설정된 값들로 보이는데 [그림 1-71]은 PHP 파일이 실행되고 난 후의 결과를 보여준다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.5/1/get_method.php?name=JisikSoft&id=Hello

[코드 1-40] Get 방식으로 전달된 데이터를 PHP 파일에서 처리하는 코드 (/1/get_method.php)

```
<?php                                                                    //---①

$name  = (isset($_GET['name'])) ? $_GET['name'] : "";                    //---②
$id    = (isset($_GET['id'])) ? $_GET['id'] : "";

$html  = '<!DOCTYPE html><html><head><title>Hello JISIKSOFT</title><meta charset="utf-
8"></meta></head><body>';                                                //---③

$html .= '<br>Your Name is "' . $name . '".<br><br>';                    //---④
$html .= 'Your Id is "' . $id . '".<br>';
$html .= '</body></html>';

print $html;                                                             //---⑤

exit();                                                                  //---⑥

?>
```

① PHP 코드는 '<?php'를 가장 위에 작성해서 PHP 언어로 동작하는 코드임을 명시하게 되며, 마지막 줄에 '?>'를 사용해서 코드의 끝을 나타낸다.

② PHP 언어에서 변수의 정의는 '$'를 변수 이름 앞에 붙여서 사용한다. 특별한 데이터 형Data Type을 정의하지 않기 때문에 JavaScript와 같이 사용이 편리하다고 볼 수 있다. 'Get 방식'으로 전달 받은 데이터는 '$_GET'이라는 PHP 언어에서 미리 정의된 예약어를 사용하는데, $_GET['name']은 name 이라는 변수로 전달받은 데이터 값을 저장하고 있다고 이해하면 된다. 'isset()' 함수는 함수 안의 매개변수가 설정되었는지를 확인하는데, "isset($_GET['name'])"과 같이 사용하면 'Get 방식'으로 name 값을 전달 받았으면 true를 반환Return하고 name 값이 존재하지 않으면 false를 반환한

다. 대부분의 프로그래밍 언어에서 사용하는 "조건 ? 참 : 거짓" 연산을 사용해서 '?' 문자 앞의 식이 참이면 $_GET(['name'])에 값이 있기 때문에 $name 변수에 값을 저장하고, 데이터를 받지 않았다면 $name 변수는 빈 문자열로 설정된다. 'Get 방식'으로 전달 받은 데이터인 name과 id 값은 이와 같은 방법으로 PHP 언어에서 받을 수 있는데, 항상 이와 같은 방법으로 사용하면 된다. 이후에 'Post 방식'으로 전달되는 값을 받는 코드를 설명하지만, 사용하는 방법은 'Get 방식'과 유사한데, '$_GET'이라는 예약어 대신에 '$_POST'라는 예약어를 사용할 뿐 값을 얻는 방법은 동일하다.

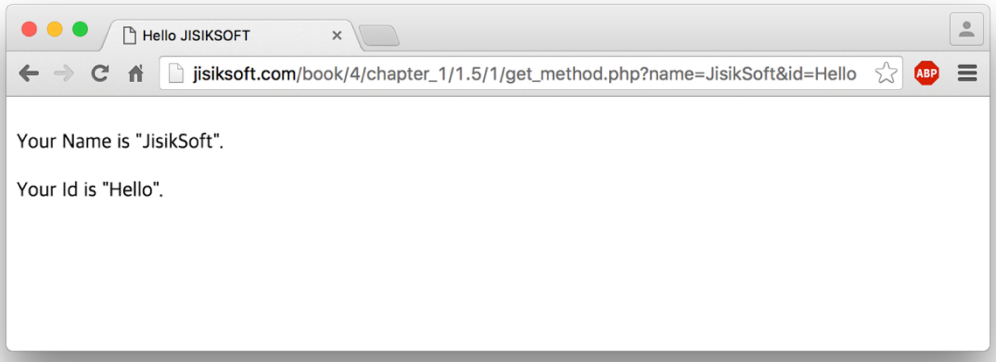
③ $html 변수는 브라우저로 보낼 내용을 저장하는 변수이며, 브라우저에서 사용하는 HTML 언어를 문자열String로 저장한다. 브라우저는 태그로 이루어진 문자열을 해석하는 기능을 갖는다는 것을 1.1장에서 이해한 독자라면 어렵게 느껴지지는 않을 것이다.

④ 브라우저에서 동작하는 JavaScript 언어는 문자열을 연결할 때 '+'(더하기) 연산자를 사용하지만, PHP에서는 '.'(마침표)를 문자열을 연결하는 연산자로 사용한다. $html 변수의 값에 연결해서 추가 문자열을 연결하는 방법을 보여주는데, '.' 연산자를 사용해서 $name 변수의 값도 문자열에 추가한다. JavaScript와 마찬가지로 문자열을 정의하는 방법은 큰따옴표(")와 작은따옴표(')를 사용한다.

⑤ print는 PHP 코드를 실행시킨 브라우저로 데이터를 보내는 예약어이며, 여기서는 $html 변수에 저장된 내용을 브라우저로 보내게 된다.

⑥ exit() 함수는 PHP의 실행을 종료하는 함수이며, 여기서는 마지막에 사용되었지만 복잡한 코드에서는 에러Error 발생 시 프로그램을 종료하기 위해서 사용할 때가 많다.

그림 1-71 Get 방식을 사용해서 데이터를 서버로 보낸 이후에 받은 결과



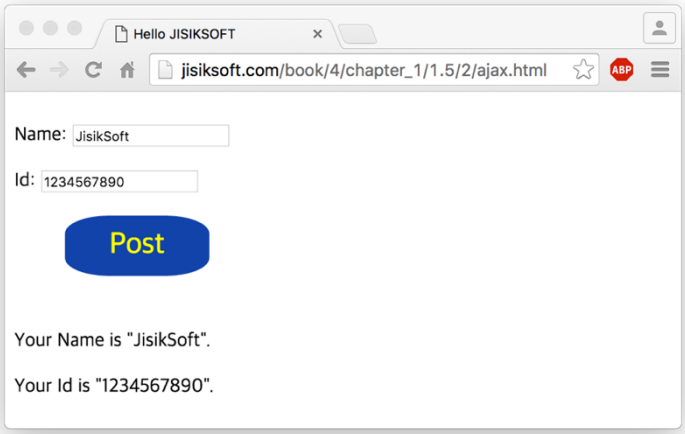
서버로 데이터를 보내는 두 가지 방법 중에 'Get 방식'은 주소 창에 직접 입력하거나 링크Hyper Link를 사용해서 사용자의 데이터를 처리한 후 보내게 하는 방법을 사용할 수 있다. 링크를 사용하는 방법은 사이트 주소로 사용하는 도메인Domain을 문자열로 처리해서 데이터를 뒤에 붙인 후 URL을 요청하는 방법으로 구현이 가능한데, 여기서는 다루지 않는다. 왜냐하면, 'Get 방식'은 새로운 페이지를 브라우저에서 여는 방법이기 때문에 현재의 웹 프로그래밍에서는 좋아하지 않는 방법이다. 위에서 설명한 'Get 방식'은 웹 프로그래밍에서의 하나의 방법이기 때문에 설명한 것이고, 이제부터 다루는 'Post 방식'을 독자분들이 항상 사용하기를 권한다. 웹 프로그래밍을 할 때 필요한 경우에는 'Get 방식'을 사용하겠지만, 필자는 'Get 방식'을 거의 사용하지 않고 프로그래밍을 진행한다. 'Post 방식'으로 데이터를 서버로 보내는 방법들은 다양한데, 이 책에서는 Ajax에이젝스라는 방법 만을 설명할 것이며 Ajax만 알고 있으면 데이터를 전송하는 거의 모든 프로그래밍에서 충분하다. 프로그래밍을 한다는 것은 많은 것을 알고 진행하는 것이 아니라 내가 가장 많이 사용하는 필요한 것들만을 사용해서 최대한 만들고, 추가적인 부분들은 인터넷에서 찾아서 마무리하면 된다. 필자는 많은 것들은 좀더 쉽게 구현하는 편이지만, 인터넷이 안되는 곳에서는 프로그래밍을 할 수도 없고 하지도 않는다.

[그림 1-72]는 사용자 브라우저에서 서버로 'Post 방식'의 데이터를 보내고 결과를 받는 방법을 보여준다. 사용자 브라우저에서는 항상 Ajax를 사용해서 데이터를 보내고, 서버에서는 PHP 또는 JSP 프로그램이 동작한 후에 결과를 보내준다. Ajax를 사용하면 사용자는 현재의 페이지가 주소가 다른 새로운 페이지로 변하는 것이 아니라 현재의 페이지에서 다양한 변화를 주는 것이 가능한데, 이 책의 중간에서 설명하는 웹 사이트를 만드는 방법은 모두 Ajax만을 사용해서 구현되었다. 이전 장에서 설명한 jQuery는 ajax() 함수를 제공하는데, 사용이 간단하고 사용자가 jQuery의 ajax() 함수를 포함하는 자신만의 함수를 추가로 만들어서 간단히 사용할 수 있다. 필자는 [코드 1-42]에서 doAjax() 함수를 만들었는데, 서버와 통신을 하는 프로그램을 만들 때 오래전에 만들었던 이 함수를 복사해서 사용하고 있다.

그림 1-72 웹에서의 데이터 이동은 브라우저에서 Ajax 를 사용하여 이루어진다.

[그림 1-73]은 'Post 방식'을 이해하기 위해서 이제부터 만들게 되는 웹 페이지를 보여주는데, 입력을 받는 <input> 태그로 만들어진 입력 창Input Box에 Name과 Id 값을 넣고 'Post' 버튼을 클릭하면 버튼 아래에 서버로부터 받은 내용을 출력하는 페이지이다. 즉, 'Post' 버튼 위는 HTML에서 만들어진 내용이며, 아래는 서버와 통신을 진행한 후에 만들어지는 내용이다.

그림 1-73 Post 방식을 사용하면 현재의 페이지를 유지하면서 페이지의 내용만을 변경할 수 있다.



[코드 1-41]과 [코드 1-42]는 'Post 방식'을 위해서 Ajax를 사용해서 진행하기 위한 브라우저에서의 코드이며, [코드 1-43]은 'Post 방식'으로 받은 데이터를 서버에서 처리한 후 결과를 재전송하는 PHP 코드다. 실제 Ajax는 'Get 방식'과 'Post 방식'을 모두 사용할 수 있게 만들어졌지만, 'Post 방식'으로만 사용하는 것이 합리적이라고 필자는 생각한다.

[코드 1-41] Post 방식을 사용하기 위하여 만들어진 HTML 코드 (/2/ajax.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./css/ajax.css"></link>
        <script src="./js/jquery-2.1.3.min.js"></script>
    </head>
    <body>
        <br>Name:
        <input type="text" id="name"><br>                    //---①
        <br>Id:
        <input type="text" id="id"><br><br>
```

```
        <div id="button">Post</div><br><br><br><br>                    //---②
        <div id="message"></div>                                       //---③

        <script src="./js/ajax.js"></script>                           //---④
    </body>
</html>
```

① <input> 태그는 사용자가 텍스트를 입력할 수 있는 입력 창Input Box을 만드는데, </input>이라는 종료 태그End Tag를 사용해도 되지만 <input> 태그만을 사용하는 것이 일반적인 규칙이다. 'name'이라는 id 값을 가진 입력창과 'id'라는 id 값을 가진 두 개의 입력 창이 만들어진다.

② 'Post'라는 버튼을 만들기 위해서 태그를 사용했으며, js 파일에서 jQuery의 click() 함수를 사용해서 동작을 구현했다.

③ 'message'라는 id 값을 갖는 태그는 서버와의 통신으로 받은 값을 넣기 위해서 만들어졌다.

④ Ajax를 사용해서 'Post 방식'으로 데이터를 보내고 결과를 받아서 'message'라는 id 값을 갖는 태그에 결과값을 넣어주는 동작을 하는 JavaScript 파일을 불러온다.

[코드 1-42] Ajax 를 사용해서 Post 방식으로 데이터를 서버로 전달하는 JavaScript 코드 (/2/ajax.js)

```
$("#button").hover(function() {
                        $( this ).css({"background-color":"purple"});
                        }, function() {
                        $( this ).css({"background-color":"#1646a7"});
                       });

$("#button").click(function() {                                        //---①
    var aName = $("#name").val();                                      //---②
    var aId = $("#id").val();
    var param = { name:aName, id:aId };                                //---③
    var result = doAjax('./post_method.php', param);                   //---④
    $("#message").empty().html(result);                                //---⑤
});

function doAjax(strUrl, inputData) {                                  //---⑥
    var result;

    $.ajax({                                                           //---⑦
```

```
            url: strUrl,
            type: 'post',
            async: false,
            datatype: 'json',
            data: inputData,
            error: function() {
                alert('Use Chrome or FireFox instead of Explorer.');
            },
            success: function(obj) {                                    //---⑧
                result = obj;
            }
        });
        return result;
}
```

① $("#button")으로 버튼 객체를 가져와서 click() 함수를 사용해서 버튼의 동작을 구현했다.

② jQuery의 val() 함수는 값Value을 갖는 태그들의 값을 가져오거나 변경할 때 사용하는 함수이며, 여기서는 입력 창을 만든 <input> 태그의 값을 가져온다. aName 변수는 'name'이라는 입력 창의 입력된 값을 저장하며, aId는 'id'라는 입력 창의 입력된 값을 갖게 된다.

③ param 변수는 'Post 방식'으로 전달하는 데이터들을 정의하는데, name이라는 값에 aName을 대입하고 id라는 값에 aId를 넣게 된다. '{ name:aName, id:aId }'와 같이 중괄호를 사용해서 값을 넣는 방법을 'JSON제이슨 방식'이라고 하는데, 이후에 좀 더 자세하게 설명할 예정이다. JavaScript에서 대괄호([,])는 배열을 선언하는데 사용하고, 중괄호({, })는 'JSON 방식'의 정렬을 위해서 사용한다.

④ 'Post 방식'으로 통신을 하기 위해서 만든 doAjax() 함수를 실행하는데, 첫 번째 매개변수인 './post_method.php'는 실행할 PHP 파일이고 두 번째 매개변수는 전달할 JSON 데이터이다. 실행할 파일의 위치는 상대경로를 사용해서 접근하는 것이 좋으며, '.'(마침표 하나)는 현재 폴더Directory를 의미하고 '..'(마침표 두 개)는 이전 폴더로 접근할 때 사용한다. Ajax 통신이 이루어지고 받은 결과는 result 변수에 저장된다.

⑤ Ajax 통신을 해서 서버로부터 받은 결과를 저장하는 result 변수의 값을 'message'라는 id 값을 가진 객체에 넣는다. 'Post' 버튼이 클릭되면 버튼 아래에 텍스트가 써지는데, 여기서 객체에 내용을 넣기 때문이며 함수가 종료된다.

⑥ jQuery의 ajax() 함수의 사용을 좀더 쉽게 하기 위해서 만든 함수가 doAjax() 함수이며, 서버에서 실행할 PHP 파일명을 포함한 인터넷 주소를 첫 번째 매개변수에서 받으며, 'Post 방식'으로 전달할 데이터는 두 번째 매개변수로 받는다.

⑦ jQuery에서는 특정 객체로 접근할 때 $() 함수 안에 객체의 클래스나 id 이름을 사용한다. 그러나, ajax() 함수와 같이 특정 객체로 접근할 필요가 없는 일반적인 jQuery 함수를 사용할 때는 $만을 사용해서 함수를 실행하는데 $.ajax()와 같은 방법으로 jQuery의 ajax() 함수를 실행할 수 있다. jQuery의 ajax() 함수는 JSON 데이터를 매개변수를 갖는데, url은 서버의 인터넷 주소를 의미하고, type은 'Post 방식'으로 데이터를 저장한다는 의미로 'post' 값을 넣었다. async는 데이터의 통신 방법을 의미하는데, 대부분의 데이터 통신 방법은 'Sync'와 'Async'(Sync의 반대말)로 구분한다. 'Sync'는 데이터를 받을 때까지 상대방으로부터 기다리고 다음 동작을 하는 것이고, 'Async'는 상대방으로 데이터를 받기 전에 다음 동작들을 수행하는 것을 의미한다. 여기서는 'Sync'를 사용하기 위해서 'async'를 'false'로 설정했는데, 에러Error를 받거나 데이터를 서버로부터 받아서 성공Success한 이후에 ajax() 함수를 종료하기 위해서다. 'Post 방식'으로 넘겨지는 데이터는 'JSON 방식'으로 만들어졌기 때문에 datatype을 'json'으로 설정했으며, data는 서버로 보내는 데이터를 의미한다. ajax() 함수가 실행되고 서버로부터 결과를 정상적으로 받으면 success 함수가 실행되고, 문제가 발생하면 error 함수가 실행된다. 1.3장의 JavaScript에서 함수를 정의하는 다양한 방법을 이해했으면, ajax() 함수에서 error와 success는 함수를 의미하고, 나머지는 변수로 사용하고 있음을 이해할 수 있을 것이다.

⑧ 서버와의 통신이 원활히 이루어지면 success() 함수가 실행되며 서버로 받은 데이터는 obj라는 매개변수가 받는다. 결과 데이터는 result 변수에 저장되고 ajax() 함수가 종료된다. 이후에 doAjax() 함수는 마지막으로 result를 함수의 결과로 전달Return하고 종료된다. error() 함수와 success()는 대부분의 프로그래밍에서 가장 어려워하는 부분일 수 있는데, 이번 장의 마지막 부분에서 설명하는 Callback콜백 함수를 이해해야 한다.

[코드 1-43] Post 방식으로 데이터를 받아서 처리하는 PHP 코드 (/2/post_method.php)

```php
<?php

$name  = (isset($_POST['name'])) ? $_POST['name'] : "";                    //---①
$id    = (isset($_POST['id'])) ? $_POST['id'] : "";

$html = '<br>Your Name is "' . $name . '".<br><br>';                       //---②
$html .= 'Your Id is "' . $id . '".<br>';

print $html;
```

```
exit();

?>
```

① 'Get 방식'에서 $_GET을 사용한 것과 마찬가지로 'Post 방식'에서는 $_POST라는 PHP 예약어를 사용해서 서버에서 값을 받는다. 'Get 방식' 코드에서 자세히 설명하였기 때문에 여기서는 코드설명은 생략하며, $name 변수는 'name'으로 전달받은 데이터를 저장하고 $id 변수는 'id'로 전달받은 데이터를 저장한다.

② 사용자 브라우저로 결과값으로 보낼 문자열String을 $html 변수에 만들어서 저장한 후에, print를 사용해서 브라우저로 전송한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.5/2/ajax.html

그림 1-74 HTML 파일을 사용해서 Post 방식으로 데이터를 보내는 초기 화면

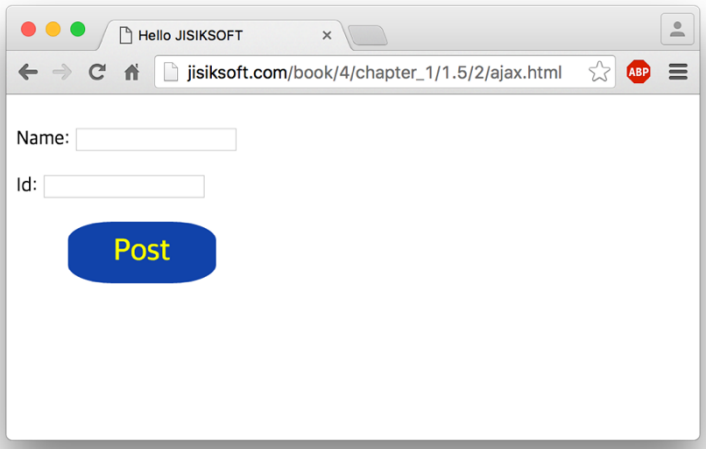
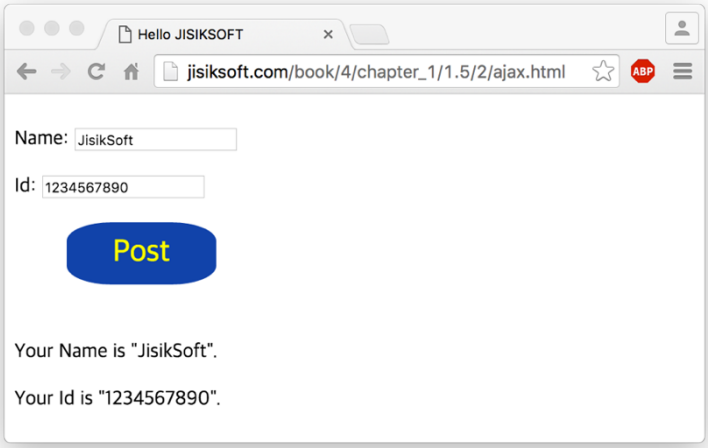


그림 1-75 Post 방식으로 데이터를 서버로 보낸 이후에 받은 결과



'Get 방식'과 'Post 방식'은 웹 프로그래밍에서 서버로 전송하는 데이터를 보내는 두 가지 방법인데, 브라우저의 주소 창에 데이터 정보가 노출되느냐로 이해하면 된다.

주소 창에 정보가 노출되는 것은 검색 서비스 이외의 사이트에서는 일반적으로 좋아하지 않기 때문에 이 책에서 앞으로 사용하는 방법은 모두가 'Post 방식'으로 진행하며, Ajax만을 사용할 예정이다. Ajax를 사용할 때 JSON 데이터를 보았는데, JSON은 JavaScript에서 즐겨 사용하는 데이터 표기방법이다. JSON 홈페이지(http://json.org)를 방문하게 되면 정규식(Regular Expression)으로 사용 방법을 설명했는데, 대학교 컴퓨터 관련학과에서 한과목으로 배우는 정규식을 깊이 이해하기 위해서 시간을 갖는 것 보다는 JSON 예제를 보는 것이 효과적일 수 있다. [그림 1-76]은 JSON 데이터에 대해서 간단히 설명하였는데, 그림의 중간에 나오는 예제 코드식을 보고 이해하는 것이 좋다. JSON은 배열과 전혀 다른 개념이고 데이터에 이름을 부여하는 부분이 생소할 수 있지만, 배열처럼 값을 나열할 수 있다는 것은 상당히 중요하다. 고급 웹 프로그래밍에서는 브라우저와 서버 간에 실시간으로 많은 데이터가 오고갈 수 있는데, JSON 표기방법을 사용하면 데이터의 길이에 상관 없이 다양한 데이터들을 주고 받는데 상당히 편리한 방법이다. JavaScript 언어에서는 JSON 데이터도 하나의 객체Object로 분류하는데, JSON 데이터 안에 있는 값들을 순차적으로 접근하는 것이 가능하기 때문이다. 실제, JSON 데이터를 alert() 함수로 확인해 보면 문자열이 아닌 'object'로 메시지 창에 나타난다.

[그림 1-76]의 마지막에는 퀴즈를 만들었는데, 예제 코드에서 퀴즈의 정답을 찾기를 바란다.

그림 1-76 JSON 데이터를 만드는 규칙 설명과 예제

<JSON 데이터>

{ } - JSON 데이터는 중괄호로 시작하고 끝난다.
[ ] - 대괄호는 배열을 선언할 때, 사용한다.
, - 다수의 데이터를 연속으로 나열할 때는 콤마(,)를 사용한다.
: - 콜론(:)의 왼쪽에는 이름(name)이고, 오른쪽은 값(value)을 의미한다.
중요한 것은 값(value)에 배열이나 JSON 데이터의 사용도 가능하다.

<JSON 예제 코드>
```
var object = {"chart1":[{"score":67},{"score":35}],"chart2":[{"score":89},{"score":72}]};
```

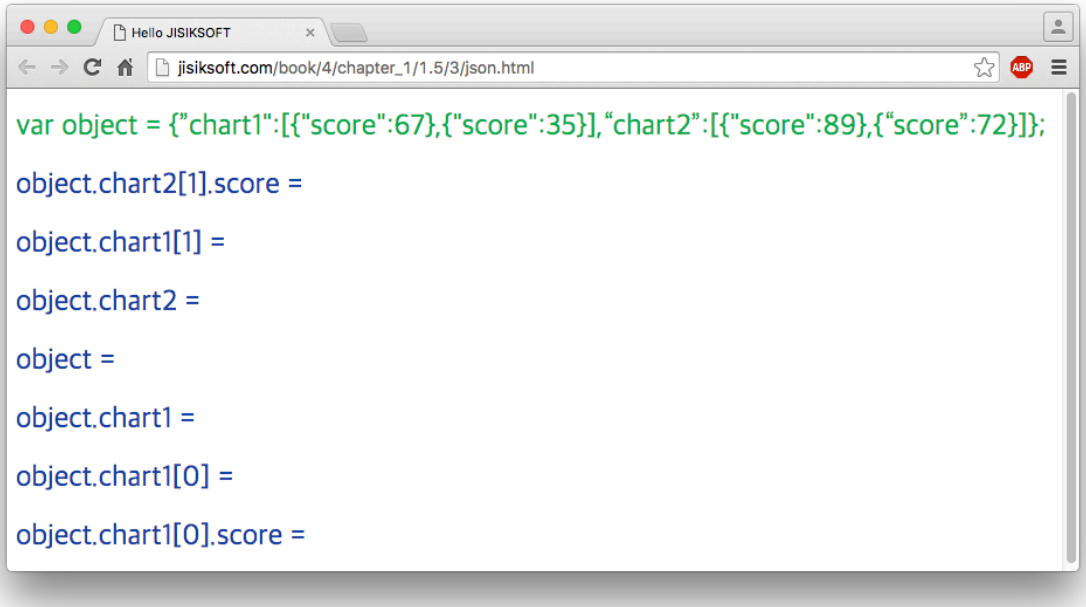
<JavaScript로 JSON 데이터에 접근하는 방법>
object : JSON 데이터로서 JSON 객체Object라고 부를 수 있다.
object.chart1 : chart1의 값은 배열인 [{"score":67},{"score":35}] 값을 갖는다.
object.chart1[0] : chart1[0]는 배열의 첫번째인 {"score":67} JSON 데이터를 갖는다.
object.chart1[0].score : chart1[0]는 배열의 첫번째에 위치한 score의 갑인 67을 의미한다.

<퀴즈>
object.chart2[1].score 의 값은?
object.chart1[1] 의 값은?
object.chart2 의 값은?

JSON 데이터를 좀더 이해하기 위해서 [그림 1-76]의 퀴즈 정답을 웹에서 보여주기 위해서 [그림 1-77]과 같은 웹 페이지를 만들었는데, [코드 1-44]와 [코드 1-45]는 웹 페이지를 만들기 위한 코드의 내용이다. 웹 프로그래밍을 하기 위해서는 반드시 JSON 데이터를 이해해야 하는데, 서버에서 운영되는 PHP 또는 JSP 프로그램과 주고 받는 데이터 형태로 가장 많이 사용되기 때문이다. 또한 1.3장에서 JavaScript 언어로 클래스를 만드는 방법을 설명하였는데, 클래스를 만들 때 사용했던 prototype은 JSON 데이터 형태로 정의된 것이다. JavaScript 언어를 설명하면서 JSON 데이터를 포함하는 것보다 서버와의 통신에 사용되는 데이터 개념으로 JSON을 바라보는 것이 좀더 좋을 수 있기 때문에 여기에 JSON 내용을 다룬다. JavaScript 클래스에서 사용한 prototype이 JSON 데이터 형태라면, JavaScript 언어에서는 JSON 데이터의 값Value으로 함수도 가질 수 있다는 것을 이해할 수 있으며, 기본 개념의 JSON 데이터보다 JavaScript 언어가 한 단계 더 발전한 JSON 데이터 형태를 사용하고 있음을 이해할 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.5/3/json.html

그림 1-77 JSON 퀴즈를 실행시킨 화면이며 각각의 식을 클릭하면 결과가 나온다.



[코드 1-44]는 [코드 1-45]는 이전 장에서 변수의 특징을 이해하기 위해서 만들었던 [코드 1-35], [코드 1-36]과 많은 부분이 동일하게 구현되었다. 즉, 태그

들을 사용해서 퀴즈를 표현하는 문자열들을 만들었으며, 각각의 문자열들을 클릭하면 결과가 오른쪽에 나온다.

[코드 1-44] JSON 퀴즈 페이지를 만들기 위한 HTML 코드 (/3/json.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./css/json.css"></link>
        <script src="./js/jquery-2.1.3.min.js"></script>
    </head>

    <body>
        <div class="expression">                                                    //---①
            var object = {”chart1":[{"score":67},{"score":35}],“chart2”:[{"score":89},{“score”:72}]};
        </div>
        <div class="variable" id="variable_1"> object.chart2[1].score = </div>     //---②
        <div class="variable" id="variable_2"> object.chart1[1] = </div>
        <div class="variable" id="variable_3"> object.chart2 = </div>
        <div class="variable" id="variable_4"> object = </div>
        <div class="variable" id="variable_5"> object.chart1 = </div>
        <div class="variable" id="variable_6"> object.chart1[0] = </div>
        <div class="variable" id="variable_7"> object.chart1[0].score = </div>

        <script src="./js/json.js"></script>                                        //---③

    </body>
</html>
```

① JSON 데이터 예제 식을 브라우저에 출력하고 CSS 디자인 적용을 위해서 “expression”이라는 클래스 이름을 가진 태그를 만들었다.

② 7개의 퀴즈들을 나타내기 위해서 7개의 태그들을 만들었으며, 각각의 태그들은 고유한 id 값을 갖고 이후에 jQuery의 click() 함수를 사용해서 퀴즈들이 클릭되었을 때 결과를 보여준다.

③ 퀴즈들이 클릭되었을 때 결과가 오른쪽에 나타나게 만드는 JavaScript 코드를 구현한 ‘json.js’ 파일이 마지막에 추가된다.

[코드 1-45] JSON 퀴즈의 결과를 보여주기 위한 JavaScript 코드 (/3/js/json.js)

```
var object = {"chart1":[{"score":67},{"score":35}], "chart2":[{"score":89},{"score":72}]};    //---①
```

```
$("#variable_1").click(function() {                                    //---②
        var result = object.chart2[1].score;
        $(this).append(result);
 });
$("#variable_2").click(function() {
        var result = object.chart1[1];
        result = JSON.stringify(result);                               //---③
        $(this).append(result);
 });
$("#variable_3").click(function() {
        var result = object.chart2;
        result = JSON.stringify(result);
        $(this).append(result);
 });
$("#variable_4").click(function() {
        var result = object;
        result = JSON.stringify(result);
        $(this).append(result);
 });
$("#variable_5").click(function() {
        var result = object.chart1;
        result = JSON.stringify(result);
        $(this).append(result);
 });
$("#variable_6").click(function() {
        var result = object.chart1[0];
        result = JSON.stringify(result);
        $(this).append(result);
 });
$("#variable_7").click(function() {
        var result = object.chart1[0].score;
        $(this).append(result);
 });
```

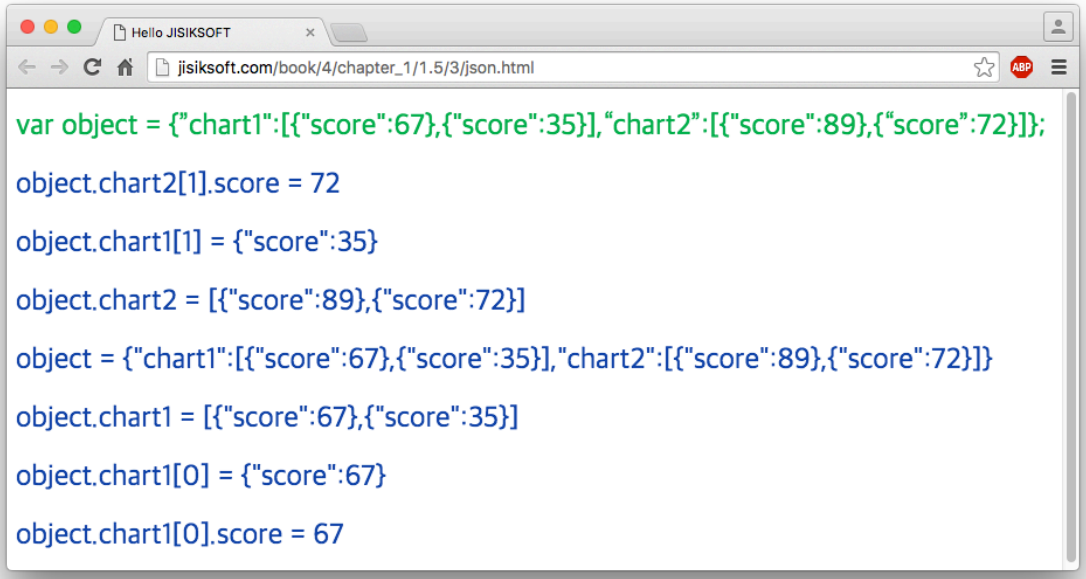
① object 변수는 JSON 데이터를 갖으며, 객체Object로 인식되기 때문에 "alert(object);"을 사용해서 값을 메시지 창에 출력하면 문자열String로 출력되지 않는다.

② 태그 객체들이 클릭되면 jQuery의 append() 함수를 사용해서 결과 값을 퀴즈의 오른쪽에 붙여넣는다.

③ JSON 데이터는 문자열String이 아니기 때문에 문자열로 변환하는 JSON.stringify() 함수를 사용해야 한다. JavaScript에서 미리 만들어진 JSON.stringify() 함수는 JSON 데이터를 확인할 때 요긴하게 사용될 수 있다. 물론 필자도 위 코드를 만들면서 함수를 찾았으며, 독자분들도 책에 나와 있는 않지만 필요한 내용들은 인터넷에서 찾아서 사용하기를 바란다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.5/3/json.html

그림 1-78 JSON 퀴즈의 식을 클릭하여 얻은 결과

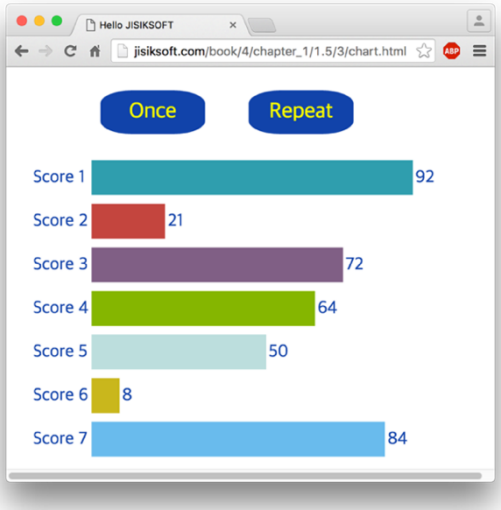


우리들이 주로 보는 일반 홈페이지들은 브라우저에서 열리게 되면 사용자의 이벤트를 받아서 페이지에 변화를 주게 된다. 하지만, 특정 소프트웨어의 모니터링 화면이나 다수가 참여하는 게임을 브라우저에서 동작하게 하려면 실시간으로 서버와 데이터 교환이 자주 발생하게 된다. 사용자 관점에서는 서버와의 통신이 자주 발생해도 혼자만 사용하는 컴퓨터이기에 무리없이 사용할 수 있지만, 서버 관점에서는 아주 많은 사용자들과 데이터를 주고 받아야 하기 때문에 주고받는 데이터의 크기가 작을수록 좋다. 그래서, 고급 웹 프로그래밍에서 추구하는 방법은 필요한 데이터만 전달하면 사용자 브라우저에서 데이터를 가공해서 사용하게 하는 것이다. 주고 받는 데이터가 작을수록 서버는 더 많은 사용자와 통신을 할 수 있기 때문에, 앞으로는 사용자 브라우저에서 더 많은 처리를 할 수 있게 JavaScript 언어를 사용해서 좀더 다양한 기능을 할 수 있게 구현하는 추세이다. 이번에 구현할 코드는 서버에 Ajax를 사용해서 데이터를 요청하면, 서버는 JSON 데이터만을 사용자 브라우저에 전달해주고 사용자 브라우저에서 데이터만을 받아서 처리하는 간단한 기능을 보여준다.

[그림 1-79]는 구현하는 웹 프로그램의 결과 페이지를 보여주는데, 'Once' 버튼은 데이터를 한 번만 요청한 후 그래프를 그려주고, 'Repeat' 버튼을 클릭하게 되면 2초마다 데이터를 받아서 새로운 그래프를 계속해서 그려준다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.5/4/chart.html

그림 1-79 데이터를 기반으로 동적으로 움직이는 그래프를 만들 수 있다.



[코드 1-46]과 [코드 1-47]은 브라우저에서 Ajax를 사용해서 서버로부터 데이터를 받아서 그래프를 그려주는 코드이며, [그림 1-48]은 서버에서 JSON 데이터를 만들어서 브라우저로 전송하는 PHP 코드이다. 전달되는 JSON 데이터는 아래와 같은 형태로 만들어지는데, JSON 데이터는 'chartData'라는 이름Name을 갖고 값Value은 7개의 JSON 데이터들을 갖는 배열로 이루어진다.

<프로그램에서 전달되는 JSON 데이터 예제>

```
{"chartData":[{"score":67},{"score":30},{"score":40},{"score":60},{"score":100},{"score":10},
{"score":98}]}
```

위와 같은 JSON 데이터는 서버에서 임의의 값을 가지고 만들게 되는데, 문자열로 만들어서 브라우저에 전달해주고 브라우저에서는 JavaScript 언어로 문자열 데이터를 JSON 데이터로 변환해서 사용한다.

[코드 1-46] 서버로부터 받은 데이터에 기반한 그래프를 그리기 위한 HTML 코드 (/4/chart.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
```

```
        <link rel="stylesheet" href="./css/chart.css"></link>
        <script src="./js/jquery-2.1.3.min.js"></script>
    </head>

    <body>
        <br>
        <div class="button" id="button_1">Once</div>                    //---①
        <div class="button" id="button_2">Repeat</div>

        <div id="chart">
            <div class="title">Score 1</div><div class="bar" id="bar_1"></div>   //---②
            <div class="title" id="score_1"></div><div class="gap"></div>
            <div class="title">Score 2</div><div class="bar" id="bar_2"></div>
            <div class="title" id="score_2"></div><div class="gap"></div>
            <div class="title">Score 3</div><div class="bar" id="bar_3"></div>
            <div class="title" id="score_3"></div><div class="gap"></div>
            <div class="title">Score 4</div><div class="bar" id="bar_4"></div>
            <div class="title" id="score_4"></div><div class="gap"></div>
            <div class="title">Score 5</div><div class="bar" id="bar_5"></div>
            <div class="title" id="score_5"></div><div class="gap"></div>
            <div class="title">Score 6</div><div class="bar" id="bar_6"></div>
            <div class="title" id="score_6"></div><div class="gap"></div>
            <div class="title">Score 7</div><div class="bar" id="bar_7"></div>
            <div class="title" id="score_7"></div>
        </div>

        <script src="./js/chart.js"></script>                           //---③

    </body>
</html>
```

① 두 개의 버튼을 태그를 사용해서 만들었다.

② 그래프는 바탕색이 다양한 사각형으로 만들면 되는데, 가로의 서버로부터 받은 JSON 데이터로부터 그래프의 가로 크기 값을 받아서 크기를 변경해 주면 된다. 그래프들의 가로 크기를 제외한 나머지는 CSS에서 디자인 속성을 미리 만들어 놓았다.

③ jQuery를 사용해서 버튼이 클릭되었을 때 동작하는 코드를 구현했으며, 그래프를 그리는 함수와 서버와의 통신을 위한 Ajax 함수는 JavaScript로 만들어졌다.

[코드 1-47] 서버로부터 받은 JSON 데이터로 그래프를 그리는 JavaScript 코드 (/4/js/chart.js)

```
var repeat;                                                             //---①

$(".button").hover(function() {
                        $( this ).css({"background-color":"purple"});
                        }, function() {
                        $( this ).css({"background-color":"#1646a7"});
```

```
                    });

$("#button_1").click(function() {                                   //---②
                        drawChart();
                        });

$("#button_2").click(function() {
                        clearInterval(repeat);                      //---③

                        drawChart();
                        repeat = setInterval(function(){ drawChart(); }, 2000);   //---④
                        });

function drawChart() {                                              //---⑤

    var jsonData = doAjax('./ajax_get_data.php');                   //---⑥

    var obj = JSON.parse(jsonData);                                 //---⑦

    for (var i=0; i<7; i++) {                                       //---⑧
        var score = obj.chartData[i].score;                         //---⑨
        var length = score * 4;                                     //---⑩

        $("#bar_"+(i+1)).css({"width":length+"px"});                //---⑪
        $("#score_"+(i+1)).empty().html(score);
    }
}

function doAjax(strUrl, inputData) {                                //---⑫

    var result;

    $.ajax({
            url: strUrl,
            type: 'post',
            async: false,
            datatype: 'json',
            data: inputData,
            error: function() {
                alert('Ajax Error.');
            },
            success: function(obj) {
                result = obj;
            }
    });
```

```
    return result;
}
```

① 화면의 'Repeat' 버튼이 클릭되면 2초마다 반복해서 서버로 데이터를 요청하기 위해서 setInterval() 함수를 사용했는데, 자동으로 수행되는 setInterval() 함수의 id 값을 저장하기 위해서 전역변수인 repeat 변수를 만들었다. 만약에 'Repeat' 버튼이 반복적으로 계속해서 클릭되면 2초마다 그래프를 그려주는 코드가 많이 실행되기 때문에, 이전에 실행되는 setInterval() 함수의 요청을 없애기 위해서 repeat 변수에 저장된 id 값을 사용해야 한다. 대부분의 프로그래밍 언어에는 주기적으로 특정 함수를 실행하는 기능들을 만들어 놓았는데, 프로그램이 종료되기 전에 주기적으로 실행되는 것을 멈추게 하기 위해서 이러한 방법을 많이 사용한다.

② 첫 번째 버튼인 'Once' 버튼이 클릭되면 그래프를 그려주는 함수를 한 번만 실행한다.

③ 두 번째 버튼인 'Repeat' 버튼이 클릭되면 2초마다 그래프를 그려주기 위해서 setInterval() 함수를 사용한다. 그런데, 만약 'Repeat' 버튼이 클릭되고 나서 다시 클릭되면 이전에 setInterval() 함수가 실행 되었기 때문에 이전 setInterval() 함수의 실행을 종료시키기 위해서 clearInterval() 함수를 사용해야 한다. 이전에 실행된 setInterval() 함수는 id 값을 반환하는데, repeat 변수에 저장된 id 값을 가지고 clearInterval(repeat) 함수를 실행해서 이전에 반복적으로 실행되는 행위를 멈추게 한다.

④ 'Repeat' 버튼이 클릭되면 drawChart() 함수를 사용해서 그래프를 한번 그려주고, setInterval() 함수를 사용해서 2초마다 drawChart() 함수를 반복해서 재실행한다.

⑤ 서버로 데이터를 요청해서 받은 데이터를 사용해서 그래프의 크기를 변경한다.

⑥ 서버로 데이터를 요청한 후 결과를 jsonData에 저장한다. 서버로 보내는 데이터가 없기 때문에 doAjax() 함수의 매개변수는 서버의 실행 파일만을 갖는다.

⑦ 서버로부터 받은 데이터는 문자열 데이터이기 때문에 JSON.parse() 예약함수를 사용해서 JSON 객체Object로 변경후 obj 변수에 저장한다. JSON.parse() 함수는 JavaScript 언어에서 제공하는 함수이다.

⑧ 데이터는 7개의 값을 갖기 때문에 for loop을 사용해서 7번 반복해서 그래프의 크기를 차례대로 변경한다.

⑨ 서버로부터 받은 JSON 데이터의 chartData는 배열로 이루어진 값이며, 배열 안의 i 번째 score 값Value을 가져와서 score 변수에 저장한다.

⑩ score 값은 0부터 100까지의 값을 갖기 때문에, 0부터 400까지의 픽셀Pixel 크기

를 갖는 그래프를 그리기 위해서 score 값에 4를 곱한다.

⑪ jQuery의 css() 함수를 사용해서 해당 그래프의 가로 크기를 변경하고, score 값을 해당 위치에 넣는다. CSS 파일에 각각의 막대 그래프의 디자인 속성이 정의되었으며, 여기서는 막대 그래프의 가로 크기만을 변경하면서 그래프에 변화를 준다.

⑫ 이 책에서 진행하는 서버와의 통신 프로그램은 항상 doAjax() 함수를 사용하는데, [코드 1-42]에서 doAjax() 함수를 자세히 설명했다.

[코드 1-48] 브라우저로 JSON 데이터를 만들어서 보내주는 PHP 코드 (/4/ajax_get_data.php)

```
<?php

$data = '{"chartData":[';                                    //---①

for ($i=0; $i<7; $i++) {                                     //---②
        if ($i != 0)
            $data .= ',';
        $data .= '{"score":' . rand(0,100) . '}';            //---③
}
$data .= ']}';

print $data;                                                 //---④
exit();

?>
```

① JSON 데이터를 만들어서 전송하기 위해서 JSON 문자열을 저장하는 $data 변수를 선언했다. JSON 데이터의 변하지 않는 부분인 '{"chartData":['를 문자열에 넣고, 이후에 추가되는 문자열들은 이전 문자열의 오른쪽에 더해진다.

② 7개의 데이터를 만들어서 $data 변수에 더하기 위해서 for loop을 사용해서 7번 반복한다. 처음에 더해지는 문자열은 앞에 ','(콤마)가 추가되지 않으며, 이후에 데이터를 추가하기 전에 배열의 구분자로서 ','를 추가한다.

③ score 이름Name을 갖는 JSON 데이터를 만드는데, 값은 0부터 100까지의 임의의 수를 만들기 위해서 PHP의 rand() 함수를 사용했다.

④ $data는 브라우저로 보내기 위한 JSON 데이터를 문자열로 저장하고 있으며, 이 데이터를 전송한 후에 exit() 함수로 PHP 프로그램을 종료한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.5/4/chart.html

그림 1-80 Once 버튼이 선택되면 서버로부터 한 번만 데이터를 받고 그래프를 그려준다.

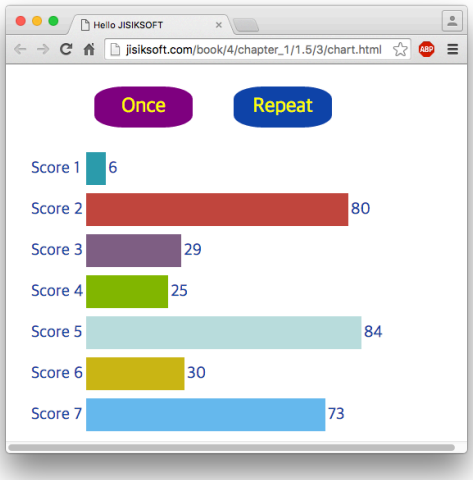
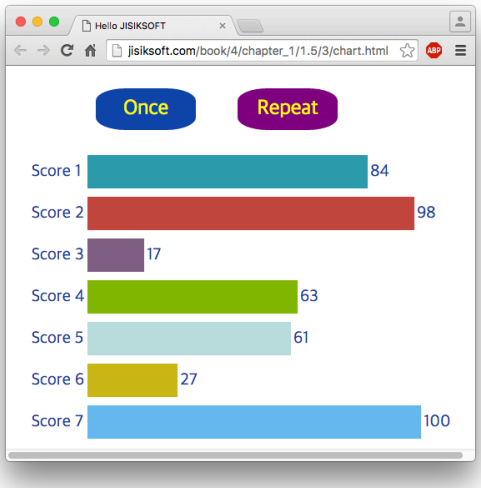


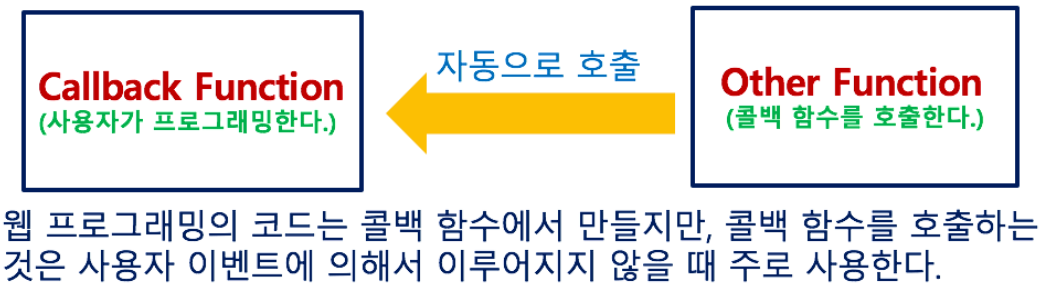
그림 1-81 Repeat 버튼을 클릭하면, 서버로 데이터를 계속 받으며 그래프를 그린다.



일반적인 프로그래밍을 하면서 많은 프로그래머가 모르고 있고, 또한 힘들어 하는 부분이 콜백(Callback) 함수의 사용이다. 저자의 두 번째 책인 "주식 분석 프로그램 만들기"의 부록에서 증권사에서 데이터를 받을 때마다 실행되는 함수가 있는데, 이것은 콜백 함수에 의해서 구현된 것이다.

[그림 1-82]는 콜백함수를 간단히 설명하기 위한 구조를 보여주는데, 콜백 함수는 프로그램을 사용하는 사용자에 의해서 실행되는 것이 아니라, 다른 곳에서 자동적으로 함수를 실행할 때 주로 사용하는 기법이다. 즉, 증권 프로그램에서는 증권사 서버에서 사용자에게 데이터를 전송하고, 사용자가 사용하는 프로그램은 데이터를 모두 받은 후에 콜백 함수가 실행되고 프로그래머가 만들어 놓은 콜백 함수안의 코드에 의해서 운영이 된다. JavaScript 언어에서도 콜백 함수는 사용자의 이벤트가 발생했을 때 실행되는 함수가 아니고, 다른 함수에서 자동적으로 콜백 함수를 호출한다고 이해하면 쉽다.

그림 1-82 많은 프로그래밍 언어에서 제공하는 콜백(Callbac) 함수의 구조

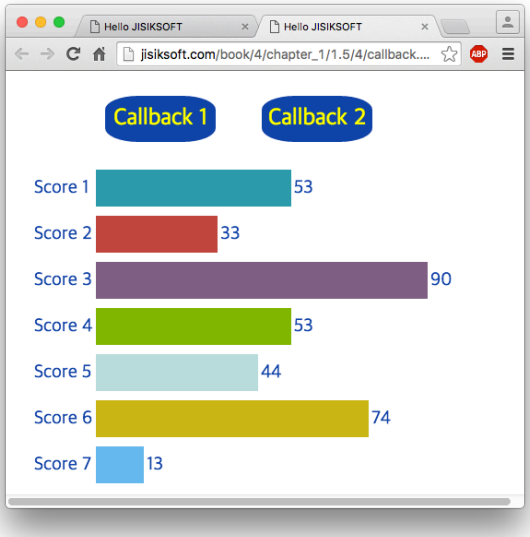


웹 프로그래밍의 코드는 콜백 함수에서 만들지만, 콜백 함수를 호출하는 것은 사용자 이벤트에 의해서 이루어지지 않을 때 주로 사용한다.

이번에 구현할 예제는 콜백 함수를 사용해서 다양한 프로그래밍이 가능하다는 것을 보여주기 위한 예제인데, 이전에 구현한 그래프 그리는 웹 프로그래밍을 콜백 함수를 사용해서 응용한 것으로 이해하면 된다. 웹 프로그래밍에서는 대부분이 서버로 데이터를 요청하고 결과 데이터를 받는 방법으로 구현되기 때문에, Ajax를 사용해서 서버로부터 데이터를 받은 후에 콜백 함수를 실행하는 방법을 구현하였다.
[그림 1-83]은 이전 웹 프로그래밍과 같은 그래프를 그려주는데, 화면에서의 변화는 두 개의 버튼 이름만이 변경되었다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.5/5/callback.html

그림 1-83 JavaScript 의 Callback 기능을 사용해서 그래프를 그려주는 프로그램 화면



이번에 만드는 웹 프로그래밍의 HTML 코드는 [코드 1-46]과 버튼 이름만 빼고 동일하기 때문에 생략했으며, [코드 1-49]는 JavaScript 언어에서의 콜백 함수를 구현하

고 [코드 1-50]은 서버에서 동작하는 간단한 PHP 프로그램을 보여준다. [코드 1-49]에서 requestData() 함수는 세 번째 매개변수로 callback이라는 이름을 가진 콜백 함수를 받는데, 이와 같이 JavaScript 언어에서는 매개변수로 함수를 전달한다는 것이 다른 프로그래밍 언어와 많이 다르다. 즉, JavaScript 언어에서는 함수의 매개변수로 함수를 사용할 수 있으며, 이 특징을 사용해서 나중에 매개변수로 받은 함수를 실행해서 콜백 기능을 하게 만들었다. JavaScript 언어에서의 콜백 함수는 코드를 보고 이해하는 것이 좋은데, [코드 1-49]의 설명에서 ④에 해당하는 requestData() 함수 호출에서 세 번째 매개변수를 function이라는 JavaScript 언어의 예약어를 사용해서 함수를 만들었다. 여기서 이해해야 할 것이 requstData() 함수를 호출할 때는 세 번째 매개변수가 함수로 정의되었지만, ⑤에서는 requestData라는 함수를 만들때 세 번째 매개변수는 callback이라는 이름을 가진 변수일 뿐이다. 이후에 ⑦에서와 같이 callback이라는 매개변수가 함수인 것을 확인한 후에 callback() 함수를 실행한다. 여기서는 callback이라는 매개변수 이름을 사용했지만, 다른 이름을 사용해서 콜백 함수를 만들 수 있다. 코드에서 Ajax를 사용하기 위해서 jQuery의 .ajax() 함수를 사용했는데, .ajax() 함수에서 error와 success는 콜백 함수인 것을 이해해야 한다. 즉, doAjax() 함수에서 $.ajax() 함수를 호출하는데, $.ajax() 함수는 서버로부터 결과를 받으면 매개변수로 전달받은 success() 함수를 호출하고 통신 장애가 발생하면 error() 함수를 실행한다. 이번 장에서 배운 JSON 데이터의 개념과 콜백 함수를 이해하였다면, $.ajax() 함수에서의 매개변수로 전달된 하나의 JSON 데이터와 JSON 데이터 안의 콜백 함수를 이해할 수 있을 것이다.

[코드 1-49] Callback 기능을 사용해서 그래프를 만드는 JavaScript 코드 (/5/js/callback.js)

```
var timer;

$(".button").hover(function() {
                        $( this ).css({"background-color":"purple"});
                        }, function() {
                        $( this ).css({"background-color":"#1646a7"});
                       });

$("#button_1").click(function() {                                    //---①
      repeat(0, 100);
});

$("#button_2").click(function() {                                    //---②
      repeat(0, 500);
```

```
});

repeat = function(min, max) {                                          //---③

        clearTimeout(timer);

        requestData(min, max, function(jsonData) {                     //---④

            var obj = JSON.parse(jsonData);

            for (var i=0; i<7; i++) {
                    var score = obj.chartData[i].score;
                    var length = score * (400 / max);

                    $("#bar_"+(i+1)).css({"width":length+"px"});
                    $("#score_"+(i+1)).empty().html(score);
            }
        });
}

requestData = function(aMin, aMax, callback) {                         //---⑤

        var param = { min:aMin, max:aMax };                            //---⑥
        var jsonData = doAjax('./ajax_get_data.php', param);

        if (callback && typeof(callback) === "function") {             //---⑦
            callback(jsonData);
        }

        timer = setTimeout(function(){ requestData(aMin, aMax, callback); }, 2000);  //---⑧
}

doAjax = function(strUrl, inputData) {                                 //---⑨

      var result;

      $.ajax({
              url: strUrl,
              type: 'post',
              async: false,
              datatype: 'json',
              data: inputData,
              error: function() {
                  alert('Ajax Error.');
              },
              success: function(obj) {
                  result = obj;
```

```
            }
    });

    return result;
}
```

① 첫 번째 버튼이 클릭되면 0부터 100사이의 그래프 값을 가지는 7개의 막대그래프를 그린다.

② 두 번째 버튼이 클릭되면 0부터 500사이의 그래프 값을 가지는 7개의 막대그래프를 그린다.

③ 그래프를 그리는 함수로서 requestData() 함수를 실행하여 서버로부터 데이터를 가져와서 콜백함수를 사용해서 7개의 그래프를 그린다. 함수의 매개변수는 두 개이며, min은 그래프의 최소값이고 max는 최대값을 갖는다. 함수 안에서 호출하는 requestData() 함수는 setTimeout()을 사용해서 주기적으로 함수를 반복실행하는데, 버튼이 클릭되면 이전에 실행된 setTimeout()을 없애기 위해서 clearTimeout() 함수를 실행한다.

④ requestData() 함수의 세 번째 매개변수로 함수를 사용했으며, 이것이 콜백함수로서 requestData() 함수에서 호출하게 된다. 콜백함수는 [코드 1-47]에서의 drawChart() 함수의 내용과 같으며, 주어진 데이터를 가지고 7개의 막대그래프들을 그린다.

⑤ requestData() 함수는 막대그래프의 범위를 매개변수로 받으며, aMin은 그래프의 최소값을 의미하고 aMax는 최대값을 갖는다. 세 번째 매개변수는 콜백함수이며 서버로부터 데이터를 받은 후에 콜백함수를 실행한다.

⑥ 서버로부터 데이터를 받기위하여 'Post 방식'으로 값의 범위를 전달하기 위하여 param 변수를 사용했으며, doAjax() 함수를 사용해서 서버와 통신을 하고 결과로 받은 데이터는 jsonData 변수에 저장된다.

⑦ 세 번째 매개변수를 받았는지 확인하고, callback 매개변수가 함수이면 callback() 함수를 실행한다. 'if(callback)'은 세 번째 매개변수를 받았을 때 true를 반환하고, 'if(typeof(callback) === "function")'은 세 번째 매개변수가 함수인지를 확인하는 조건문이다. 서버로부터 받은 결과를 callback() 함수의 매개변수로 넣어서 실행한다.

⑧ 2초마다 그래프를 그리기 위해서 setTimeout() 함수를 사용해서 requestData() 함수를 실행한다.

⑨ $.ajax() 함수는 하나의 매개변수를 넣어서 실행하는데, 매개변수는 JSON 데이터 형태를 갖으며 'error'와 'success'는 콜백함수로 사용되었다. 웹 프로그래밍에서 중요

한 것이 JavaScript 언어에서의 JSON 데이터와 콜백함수의 사용을 이해하는 것이며, jQuery의 ajax() 함수는 아주 좋은 예제이다.

[코드 1-50] (min, max) 범위의 데이터를 만들어서 보내주는 PHP 코드 (/5/ajax_get_data.php)

```
<?php

$min    = (isset($_POST['min'])) ? $_POST['min'] : "";                    //---①
$max    = (isset($_POST['max'])) ? $_POST['max'] : "";

$data = '{"chartData":[';

for ($i=0; $i<7; $i++) {
        if ($i != 0)
            $data .= ',';
        $data .= '{"score":' . rand($min, $max) . '}';                    //---②
}
$data .= ']}';

print $data;
exit();

?>
```

① 그래프의 범위의 최대값과 최소값을 'Post 방식'으로 받아서 JSON 데이터를 만들어서 브라우저로 전송한다.

② PHP 언어에서 임의의 값을 만드는 rand() 함수에 최소값과 최대값의 범위를 매개변수로 넣어서 특정 범위의 값을 얻는다.

확인 사이트: http://jisiksoft.com/book/4/chapter_1/1.5/5/callback.html

그림 1-84 'Callback 1' 버튼이 클릭되면 그래프는 최대 100 까지의 값을 갖는다.

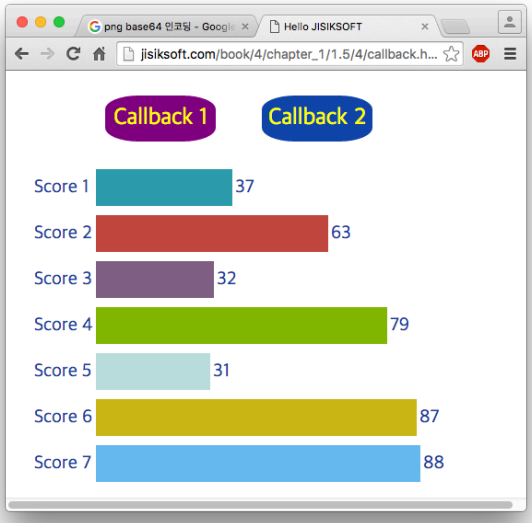
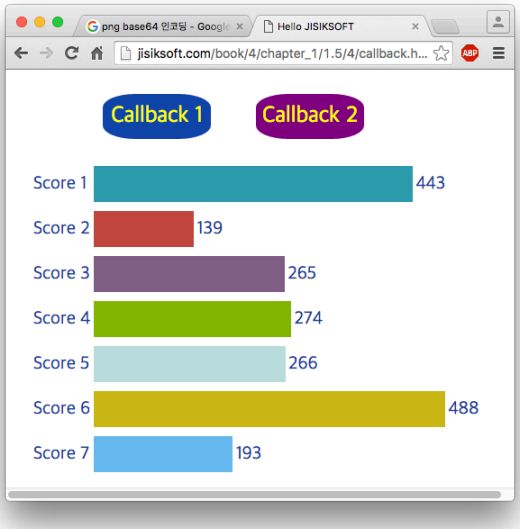


그림 1-85 'Callback 2' 버튼이 클릭되면 그래프는 최대 500 까지의 값을 갖는다.



[그림 1-84]와 [그림 1-85]는 두 개의 버튼이 클릭되었을 때의 결과을 보여주는데, 두 개의 차이는 각각의 막대그래프의 오른쪽에 나와 있는 score 값이 다르다. 이번 예제에서는 함수를 호출할 때 매개변수로 값의 범위를 정해주는 방법을 사용했는데, 많은 프로그램에서는 매개변수의 값에 따라 다른 값들을 서버로부터 받는 다양한 기법들을 사용한다.

이번 장에서는 PHP 언어를 설명하는 내용이지만, PHP 언어의 특성보다는 JavaScript 의 주요 내용이 더 많은 비중을 차지했다. PHP 언어는 서버와의 통신을 기본으로 이해해야 하는데, 많은 웹 프로그램에서 주고 받는 데이터가 JSON 데이터 구조를 가지기 때문에 이번 장에서 JSON 데이터를 설명하였다. 또한, JavaScript 언어에서 아주 중요한 콜백 함수도 서버와의 통신에 기반하기 때문에 이번 장에서 JavaScript 언어를 좀더 깊이 있게 다루었다. 저자가 만들고 있는 "How-to Series"는 프로그래밍 입문책이라기 보다는 프로그래밍 언어를 하나라도 접했던 초중급 이상의 독자들을 위한 책이기 때문에 기본 연산자에 대한 설명을 하지는 않았다. 웹 프로그래밍은 HTML, CSS, JavaScript, PHP(or JSP) 등의 모든 언어의 특징과 사용하는 이유를 알아야 하는데, 많은 책들이 4개의 언어 중 하나만을 설명하고 있을 때가 많다. 지금까지 진행된 웹 프로그래밍의 기본언어들의 특징들을 충분히 이해한 독자라면, 다음 Chapter부터 진행되는 웹 프로그래밍 예제들을 쉽게 이해할 수 있을 것이다. 또한 여기서 설명하지 않은 기법들을 응용한다면, 자신만의 웹 프로그램을 만드는데 상당

히 많은 도움이 될 것이다. 이번 장에서 PHP 언어에 대한 코드가 다양하지 않았지만, 이후의 Chapter들에서 다양한 PHP 예제들을 보게 될 것이며 자신에게 필요한 것들은 인터넷에서 직접 찾아서 프로그래밍의 방법을 익히기를 바란다. 한 회사의 홈페이지를 만드는데도 상당히 긴 시간이 필요하며 웹 프로그래밍은 화면의 픽셀 Pixel 값들을 조정하기 위해서도 많은 시간이 걸리기 때문에 조급한 마음보다는 천천히 자신의 실력을 향상시키는 것이 필요하다.

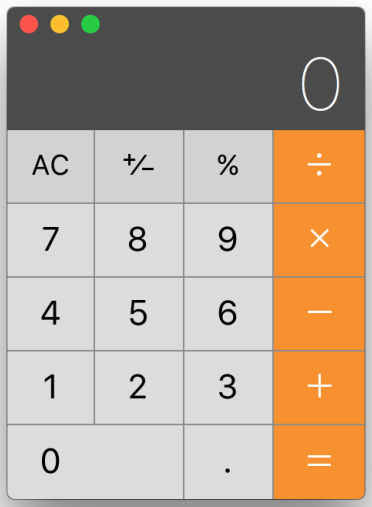
chapter 2

계산기 만들기

프로그래밍을 연습할 때, 연습으로 좋은 것은 계산기 프로그램이다. 가장 쉬운 사칙 연산을 사용해서 만들기 때문에 어려운 것이 없고, 기본적인 개념을 이해하면 누구나 쉽게 만들 수 있다.

[그림 2-1]은 애플Apple에서 만든 맥용 계산기 프로그램의 이미지다. 이번 Chapter에서는 브라우저에서 사용할 수 있는 계산기를 만드는데, HTML, CSS, JavaScript 세 가지 언어만으로 브라우저에서 사용할 수 있는 응용 프로그램을 어떻게 구현하는지 이해할 수 있을 것이다.

그림 2-1 MacBook 계산기 프로그램



간단한 웹 프로그래밍은 브라우저에서 디자인을 예쁘게 만들고, 이벤트 처리를 하면 된다고 생각하면 된다. 그래서, 디지인을 만들기 위해서 HTML과 CSS를 사용해서 브라우저에 계산기 모양을 만들 것이며, 이후에 각 버튼을 클릭하면 처리되는 이벤트를 위하여 JavaScript를 사용한다.

2.1 계산기 디자인

이번 장에서는 애플의 기본 계산기를 브라우저에서 만들기 위하여 HTML과 CSS만을 사용해서 브라우저에서 계산기를 디자인한다. HTML 언어에서는 태그만을 사용해서 내용을 넣는데, 필자는 대부분의 웹 프로그래밍을 태그만을 사용해서 만드는 편이다. 태그들의 종류는 상당히 많은데, 1.2장에서 설명한 CSS를 잘 활용하면 대부분의 웹 사이트와 웹 응용 프로그램은 태그만을 사용해서 만들 수 있다. [코드 2-1]과 [코드 2-2]는 계산기를 위한 HTML과 CSS 코드를 보여준다.

[코드 2-1] 계산기 디자인 HTML 코드 (calculator.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./css/calculator.css"></link>                    //---①
    </head>

    <body>
        <div id="calculator">                                                         //---②
            <div id="board">                                                          //---③
                <div class="row" id="row_1">                                          //---④
                    <div id="result"> 0 </div>
                </div>
                <div class="row">                                                     //---⑤
                    <div class="square operator2" id="clear"> C </div>                //---⑥
                    <div class="square operator2" id="sign"> +/- </div>
                    <div class="square operator2" id="percent"> % </div>
                    <div class="square operator1" id="divide"> / </div>
                </div>
                <div class="row">
                    <div class="square" id="no_7"> 7 </div>
                    <div class="square" id="no_8"> 8 </div>
                    <div class="square" id="no_9"> 9 </div>
```

```
                <div class="square operator1" id="multiply"> x </div>
            </div>
            <div class="row">
                <div class="square" id="no_4"> 4 </div>
                <div class="square" id="no_5"> 5 </div>
                <div class="square" id="no_6"> 6 </div>
                <div class="square operator1" id="minus"> - </div>
            </div>
            <div class="row">
                <div class="square" id="no_1"> 1 </div>
                <div class="square" id="no_2"> 2 </div>
                <div class="square" id="no_3"> 3 </div>
                <div class="square operator1" id="plus"> + </div>
            </div>
            <div class="row">
                <div class="square" id="no_0"> 0 </div>
                <div class="square" id="point"> . </div>
                <div class="square operator1" id="equal"> = </div>
            </div>
        </div>
    </div>
</body>
</html>
```

① 계산기의 디자인은 'calculator.css' 파일에 저장되었으며, 여기서 파일을 연결Link한다.

② 계산기의 전체 내용을 포함하는 'calculator' id 값을 가진 태그를 사용해서 계산기의 위치를 전체적으로 변경하는 것이 가능하다.

③ 계산기의 내용을 회로의 기판을 의미하는 'board'라는 id 값을 사용해서 모든 버튼을 감싸준다.

④ 계산기의 이미지도 하나의 표Table로 이해할 수 있으며, 행Row과 열Column의 조합으로 이해해도 된다. 첫 번째 계산의 결과값이 나오는 것은 'result'라는 id 값을 가진 태그에 표시되며 가장 첫 번째 행Row으로 구조화했다.

⑤ 계산기의 모든 버튼들은 5개의 행으로 나란히 배열되었기 때문에 'row'라는 클래스 값을 가진 태그들을 사용해서 각 행의 버튼들을 감쌌다.

⑥ 각각의 버튼들은 디자인을 위해서 클래스와 id 값들을 사용해서 태그들로 만들었으며, 이후에 이 태그들에게 onclick 이벤트를 사용해서 JavaScript 언어로 동작을 구현하게 된다.

[코드 2-2] 계산기 디자인 CSS 코드 (/css/calculator.css)

```
#calculator {                                                    //---①
    position:relative;
```

```
        left:50px;
        width:227px;
        height:317px;
        background-color:#999999;
}
#board {                                                    //---②
        position:absolute;
        top:22px;
        left:2px;
        width:223px;
        height:254px;
}
.square {                                                   //---③
        position:relative;
        width:55px;
        height:48px;
        font-size:23px;
        color:#000000;
        text-align:center;
        line-height:55px;
        background-color:#e0e0e0;
        float:left;
}
.operator1 {                                                //---④
        color:#ffffff;
        background-color:#f5923e;
}
.operator2 {                                                //---⑤
        background-color:#d6d6d6;
}
.row {
        height:49px;
}
#result {                                                   //---⑥
        font-size:50px;
        color:#ffffff;
        text-align:right;
}
##### 이하 생략 #####
```

---

① 계산기 전체의 위치는 여기서 하며, 가로와 세로의 크기를 넣어주고 바탕색을 회색으로 만들었다.

② 'board'라는 id 값을 가진 태그는 'calculator'라는 id 값을 가진 태그에 종속되기 때문에 위치Position를 'absolute'로 설정하고 위치와 크기를 정하였다.

③ 모든 버튼의 크기, 글자색, 바탕색 등을 설정했다.

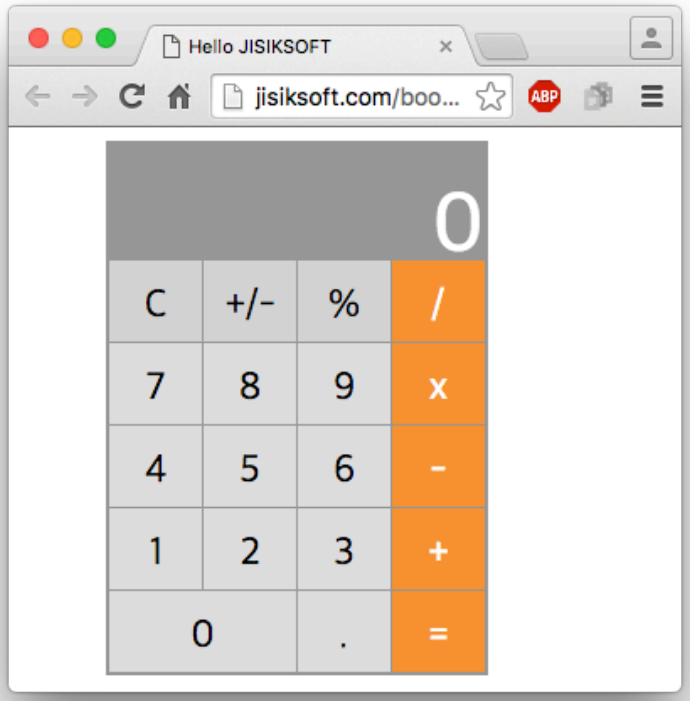
④ 계산기의 오른쪽에 있는 오렌지색 버튼의 디자인을 위해서 사용된다.

⑤ 계산기의 가장 위에 있는 세 개의 버튼을 다른 버튼보다 조금 더 진한 회색으로 변경하기 위해서 사용했다.

⑥ 계산기의 가장 위에 있는 결과값을 위해서 하얀색 글자의 크기를 정하였고, 글자를 오른쪽에 위치하도록 'text-align'을 'right'으로 설정하였다.

확인 사이트: http://jisiksoft.com/book/4/chapter_2/2.1/calculator.html

그림 2-2 HTML 과 CSS 만을 사용해서 디자인된 계산기



지금까지 구현한 것은 사용자의 이벤트를 받을 수는 없지만, 계산기의 디자인은 애플Apple사의 계산기와 비슷하게 만들어졌다. 이와 같이, HTML과 CSS 만으로도 브라우저에서 멋진 이미지를 만들 수는 있지만, 아무런 동작도 하지 않기 때문에 아직은 살아있지 않은 웹 프로그래밍이다. 다음 장에서는 onclick을 사용해서 모든 버튼이 클릭되었을 때 반응하는 것을 구현한다.

2.2 계산기 이벤트

이번 장에서는 계산기에 이벤트를 추가하는데 [코드 2-3]에서 onclick 이벤트를 사용해서 JavaScript 함수 한 개를 실행한다. 실행되는 pressBtn() 함수는 매개변수가 한 개이며, 이 매개변수의 값에 따라 각각의 버튼 기능을 수행한다. 프로그램을 만들 때는 단계가 필요한데, pressBtn()을 실행할 때, 각각의 버튼이 올바르게 동작하는지를 확인하기 위하여 alert() 함수만을 사용해서 [코드 2-4]와 같이 JavaScript 함수를 구현했다.

[코드 2-3] 계산기 이벤트 처리 HTML 코드 (calculator.html)

```
<!DOCTYPE html>

<html>
    <head>
        <title>Hello JISIKSOFT</title>
        <meta charset="utf-8"></meta>
        <link rel="stylesheet" href="./css/calculator.css"></link>
        <script src="./js/calculator.js"></script>
    </head>

    <body>
        <div id="calculator">                                                    //---①
            <div id="board">
              <div class="row" id="row_1">
                  <div id="result"> 0 </div>
              </div>
              <div class="row">
                <div class="square operator2" id="clear" onclick="pressBtn('clear');"> C </div>
                <div class="square operator2" id="sign" onclick="pressBtn('sign');"> +/- </div>
                <div class="square operator2" id="percent" onclick="pressBtn('percent');">
                                                                                  % </div>
                <div class="square operator1" id="divide" onclick="pressBtn('divide');">
                                                                                  / </div>
              </div>
              <div class="row">
                <div class="square" id="no_7" onclick="pressBtn('7');"> 7 </div>
                <div class="square" id="no_8" onclick="pressBtn('8');"> 8 </div>
                <div class="square" id="no_9" onclick="pressBtn('9');"> 9 </div>
                <div class="square operator1" id="multiply" onclick="pressBtn('multiply');">
                                                                                  x </div>
              </div>
              <div class="row">
                <div class="square" id="no_4" onclick="pressBtn('4');"> 4 </div>
                <div class="square" id="no_5" onclick="pressBtn('5');"> 5 </div>
                <div class="square" id="no_6" onclick="pressBtn('6');"> 6 </div>
```

```
                <div class="square operator1" id="minus" onclick="pressBtn('minus');">
                                                                                - </div>
            </div>
            <div class="row">
                <div class="square" id="no_1" onclick="pressBtn('1');"> 1 </div>
                <div class="square" id="no_2" onclick="pressBtn('2');"> 2 </div>
                <div class="square" id="no_3" onclick="pressBtn('3');"> 3 </div>
                <div class="square operator1" id="plus" onclick="pressBtn('plus');"> + </div>
            </div>
            <div class="row">
                <div class="square" id="no_0" onclick="pressBtn('0');"> 0 </div>
                <div class="square" id="point" onclick="pressBtn('.');"> . </div>
                <div class="square operator1" id="equal" onclick="pressBtn('equal');"> = </div>
            </div>
        </div>
    </div>
</body>
</html>
```

① 각각의 버튼에 onclick 이벤트를 사용해서 pressBtn() 함수를 실행한다.

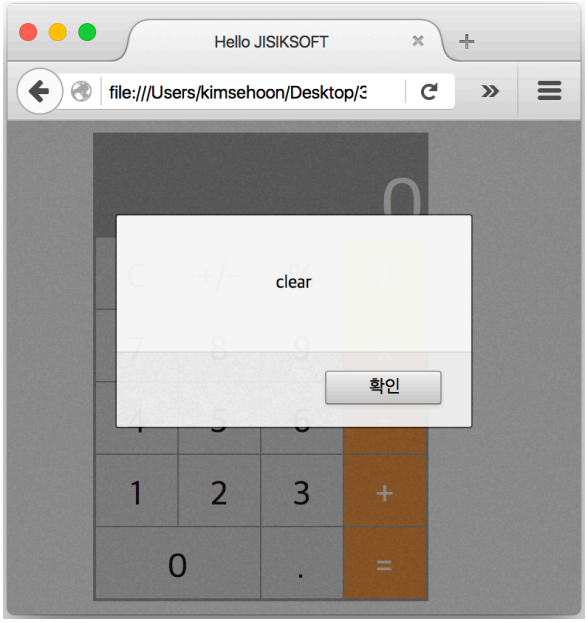
[코드 2-4] 계산기 이벤트 처리 JavaScript 코드 (/js/calculator.js)

```
pressBtn = function(input) {                                    //---①

    if (input == 'clear') {
        alert('clear');
    } else if ((input >= 0) && (input <= 9)) {
        alert('number : ' + input);
    } else if (input == '.') {
        alert('.');
    } else if (input == 'sign') {
        alert('sign');
    } else if (input == 'percent') {
        alert('percent');
    } else if (input == 'divide') {
        alert('divide');
    } else if (input == 'multiply') {
        alert('multiply');
    } else if (input == 'minus') {
        alert('minus');
    } else if (input == 'plus') {
        alert('plus');
    } else if (input == 'equal') {
        alert('equal');
    }
}
```

① 함수가 실행될 때 전달받는 매개변수 input의 내용에 따라서 클릭된 버튼의 이름이 메시지 창에 출력된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_2/2.2/calculator.html

그림 2-3 'C' 버튼이 클릭되었을 때, 'clear'란 문자가 메시지 창에 표시되었다.



프로그래밍을 할 때 함수의 구조를 먼저 만들고 내용을 순차적으로 넣어서 구현하는 것이 좋다. 하나의 타이핑Typing이 모여서 단어Word가 되고, 이것이 줄Line이 되면서 프로그래밍을 구현하게 된다. 프로그래밍 중간에 진행되는 테스트는 귀찮은 작업일지라도 결과적으로는 시간을 절약하는 아주 좋은 도구이다. 저자의 첫 번째 책인 "Big Number 연산"의 부록에 저자의 프로그래밍 방법을 기술했는데, 괜찮은 방법임을 자신하기에 독자에게 공개했다.

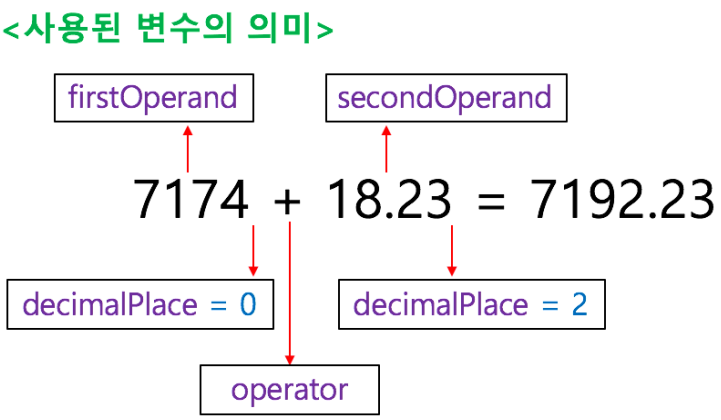
2.3 계산기 함수 구현

이번 장에서 계산기의 기능을 모두 구현하며, 이와 같은 원리로 브라우저가 아닌 다른 곳에서 동작하는 계산기를 만들 수 있다. 프로그래밍 언어에서 자유로와진다는 것은 프로그래밍의 구조를 이해하고 각 언어의 특징을 파악해서 잘 활용하는 것이다. 프로그래밍의 구조는 알고리즘을 좀더 공부한 사람들이 더 많이 이해하는 편이다. 프로그래밍에서 완벽한 프로그램이란 것은 없을 수 있다. 단지, 우리들이 더 치중해야 할 것은 더 안정적이고 빠르게 동작하는 프로그램을 만들기 위해 노력하는

것이다. 프로그램의 포장지가 디자인이라면, 웹 프로그래밍에서의 상품성은 JavaScript에 좀더 비중을 둘 수 있는데, 저자가 책에서 만든 것을 정답으로 여기지 말고 더 효율적인 프로그램을 만들 수 있다고 생각하고 접근하기를 권한다.

[코드 2-5]는 계산기에서 필요한 변수들을 전역변수로 선언했으며, 이 변수들에게 필요한 값들을 변경하고 저장하면서 계산기가 동작하게 된다. 아래의 [그림 2-4]는 사용되는 변수들의 의미를 그림으로 보여주는데 계산기에서 처리하는 수식의 값들을 각각의 변수들은 저장한다. 변수의 이름을 정할 때는 변수의 특징을 이름으로 이해할 수 있도록 정하는 것이 좋다. 즉, firstOperand는 첫 번째 숫자를 의미하고, secondOperand는 두 번째 숫자를 의미한다. 계산되는 수는 실수이기 때문에 소수점의 위치를 의미하는 decimalPlace 변수를 사용해서 정수일 때는 항상 0의 값을 갖고 소수점 뒤에 숫자가 추가될 때는 1씩 증가한다. 마지막 operator 변수는 연산자를 의미하며, 여기서는 '+', '-', '*', '/' 네 개의 값 중에 하나를 저장한다.

그림 2-4 코드에서 사용되는 변수들의 의미



[코드 2-5]는 각각의 버튼이 클릭되었을 때마다 pressBtn() 함수가 실행되면서, 처리되는 값들을 전역변수에 저장하고 계산기 화면에 결과를 보여준다.

[코드 2-5] 계산기 JavaScript 함수 구현 (/js/calculator.js)

```
var decimalPlace = 0;                                    //---①
var firstOperand = 0;                                    //---②
var secondOperand = null;                                //---③
var operator = null;                                     //---④

pressBtn = function(input) {
```

```
if (input == 'clear') {                                          //---⑤

    decimalPlace = 0;
    firstOperand = 0;
    secondOperand = null;
    operator = null;
    adjustResultSize(firstOperand);                              //---⑥
    $("#result").empty().html(firstOperand);                     //---⑦

} else if ((input >= 0) && (input <= 9)) {                       //---⑧

    if (operator == null) {                                      //---⑨
        if (decimalPlace == 0) {
            firstOperand = (firstOperand * 10) + (input * 1);
        } else {                                                 //---⑩
            var temp = 1;
            for (var i=0; i<decimalPlace; i++) {
                temp = temp * 0.1;
            }
            firstOperand = firstOperand + (input * temp);
            decimalPlace += 1;
        }
        adjustResultSize(firstOperand);
        $("#result").empty().html(firstOperand);
    } else {                                                     //---⑪
        if (decimalPlace == 0) {
            secondOperand = (secondOperand * 10) + (input * 1);
        } else {
            var temp = 1;
            for (var i=0; i<decimalPlace; i++) {
                temp = temp * 0.1;
            }
            secondOperand = secondOperand + (input * temp);
            decimalPlace += 1;
        }
        adjustResultSize(secondOperand);
        $("#result").empty().html(secondOperand);
    }

} else if (input == '.') {                                       //---⑫

    if (decimalPlace == 0)
        decimalPlace = 1;

} else if (input == 'sign') {                                    //---⑬
```

```
        if (operator == null) {
            firstOperand = firstOperand * '-1';
            adjustResultSize(firstOperand);
            $("#result").empty().html(firstOperand);
        } else {
            secondOperand = secondOperand * '-1';
            adjustResultSize(secondOperand);
            $("#result").empty().html(secondOperand);
        }

    } else if (input == 'percent') {                                    //---⑭

        if (operator == null) {
            firstOperand = firstOperand * 0.01;
            adjustResultSize(firstOperand);
            $("#result").empty().html(firstOperand);
        } else {
            secondOperand = secondOperand * 0.01;
            adjustResultSize(secondOperand);
            $("#result").empty().html(secondOperand);
        }

    } else if (input == 'divide') {                                     //---⑮

        operator = '/';
        secondOperand = 0;
        decimalPlace = 0;

    } else if (input == 'multiply') {

        operator = '*';
        secondOperand = 0;
        decimalPlace = 0;

    } else if (input == 'minus') {

        operator = '-';
        secondOperand = 0;
        decimalPlace = 0;

    } else if (input == 'plus') {

        operator = '+';
        secondOperand = 0;
        decimalPlace = 0;

    } else if (input == 'equal') {                                      //---⑯

        if (operator == null) {
```

```
                return;
            } else if (operator == '/') {
                if (secondOperand == 0)
                    return;
                firstOperand = firstOperand / secondOperand;
            } else if (operator == '*') {
                firstOperand = firstOperand * secondOperand;
            } else if (operator == '-') {
                firstOperand = firstOperand - secondOperand;
            } else if (operator == '+') {
                firstOperand = firstOperand + secondOperand;
            }
            adjustResultSize(firstOperand);
            $("#result").empty().html(firstOperand);
            decimalPlace = checkDecimalPlace(firstOperand);                    //---⑰
            secondOperand = null;
            operator = null;
        }
}

checkDecimalPlace = function(input) {                                          //---⑱
    var num = input + '.';
    var value = num.split('.')[1].toString().length;
    return value;
}

adjustResultSize = function(input) {                                           //---⑲
    var len = input.toString().length;
    if (len < 9) {
        $("#result").css("font-size","50px");
    } else if (len < 10) {
        $("#result").css("font-size","45px");
    } else if (len < 11) {
        $("#result").css("font-size","40px");
    } else if (len < 12) {
        $("#result").css("font-size","37px");
    } else if (len < 13) {
        $("#result").css("font-size","34px");
    } else if (len < 14) {
        $("#result").css("font-size","31px");
    } else if (len < 15) {
        $("#result").css("font-size","28px");
    } else if (len < 16) {
        $("#result").css("font-size","26px");
    } else {
        alert("계산할 수 없습니다.");
    }
}
```

① 소수점의 위치를 저장하는 변수이며, 정수의 값을 갖을 때는 항상 0의 값을 갖

는다. 만약 소수점을 의미하는 '.'(마침표)가 클릭되면 이후부터 1씩 증가는데, 소수점 오른쪽으로 숫자가 추가될 때마다 decimalPlace 변수에 1씩 더해진다.

② firstOperand 변수는 계산을 하기 위한 첫 번째 값을 저장한다.

③ secondOperand 변수는 계산을 하기 위한 두 번째 값을 저장한다.

④ operator 변수는 연산을 하기 위한 4개의 사칙 연산자(+, -, *, /)를 저장하면 null로 초기화된다. 만약 operator가 null이면 숫자가 클릭될 때마다 firstOperand 변수의 값이 변경되고, 연산자를 저장하고 있으면 secondOperand 변수의 값이 변경된다.

⑤ 'C'(clear) 버튼이 클릭되면 모든 변수들은 초기화되고, 계산기의 결과값도 0으로 변경한다.

⑥ 계산기의 결과값은 숫자가 특정 갯수 이상으로 많아지면 글자의 크기가 변경되는데, 이것을 처리하기 위해서 adjustResultSize() 함수를 사용한다.

⑦ 계산기의 결과 화면에 결과를 출력할 때는 'result' id 값을 가지고 있는 객체의 내용을 변경하면 된다.

⑧ 0부터 9까지의 숫자가 클릭되면 연산자가 이전에 클릭되었는지를 확인하는데, 만약 연산자가 클릭되지 않았다면 operator가 null의 값을 갖기 때문에 firstOperand 변수의 값을 가지고 처리한다. 이와 같은 방법으로 연산자가 이전에 클릭되었다면 operator 값이 null이 아니기 때문에 secondOperand 변수의 값을 변경하면 된다.

⑨ 숫자가 클릭되기 이전에 연산자 버튼이 클릭되지 않았다면 operator 값이 null이기 때문에 firstOperand 변수의 값에 입력되는 숫자를 넣는다. 'decimalPlace' 변수가 0이면 정수값을 갖기 때문에 firstOperand에 10을 곱해주고 클릭된 숫자(input)의 값을 더해준다.

⑩ 숫자가 클릭되기 전에 '.'(마침표)가 클릭되었다면 소수점을 가진 실수로 변경되기 때문에 decimalPlace는 0보다 큰 값을 갖는다. 실수라면 현재 클릭된 숫자는 소수점 오른쪽에 더해지게 되는데, decimalPlace 값에 기반해서 for loop을 사용하여 0.1을 반복해서 곱한 후 temp 변수에 저장한다. 이후에 현재 클릭된 숫자를 temp 값에 곱하고 firstOperand에 더해주면 소수점 오른쪽에 숫자가 추가된다.

⑪ 이전에 연산자(+,-,*,/) 버튼이 클릭되었다면 operator 값은 null이 아니기 때문에 secondOperand 변수의 값을 가지고 클릭되는 숫자를 처리하게 된다. 숫자가 처리되는 과정은 firstOperand에서 처리된 과정과 동일하고 단지 secondOperand 변수로만 변경되었다.

⑫ '.'(마침표)가 클릭되면 현재 숫자가 정수이면 decimalPlace 변수가 0이기 때문에 실수로 계산하기 위해서 decimalPlace 값을 1로 변경한다.

⑬ '+/-' 버튼이 클릭되면 양수는 음수로 변경되고, 음수는 양수로 변경된다. 해당 값에 '-1'을 곱해주기만 하면 되며, operator가 null이면 firstOperand 변수의 값을 변경하고 null이 아니면 secondOperand 변수의 값을 변경한다.

⑭ '%' 버튼이 클릭되면 0.01을 곱해주는데, ⑬에서와 같은 방법으로 진행되며 단지 곱해주는 값이 0.01로 변경되었을 뿐이다.

⑮ 사칙 연산자가 클릭되면 operator 값에 연산자를 넣어주고, 값이 존재하지 않는 의미로 null 값을 가지고 있던 secondOperand 변수를 0으로 변경한다. 또한, 이후부터는 secondOperand 변수를 가지고 숫자가 처리되기 때문에 decimalPlace도 0으로 초기화된다.

⑯ '=' 버튼이 클릭되면 연산자에 기반하여 연산을 수행하게 되는데, operator가 null이면 연산자가 없기 때문에 아무런 동작도 하지 않으며, 또한 나누기(/) 연산을 할 때는 나누는 값이 0이어도 안된다. 연산이 진행된 이후에 결과값은 firstOperand에 저장되고 secondOperand와 operator 변수는 null로 초기화한다.

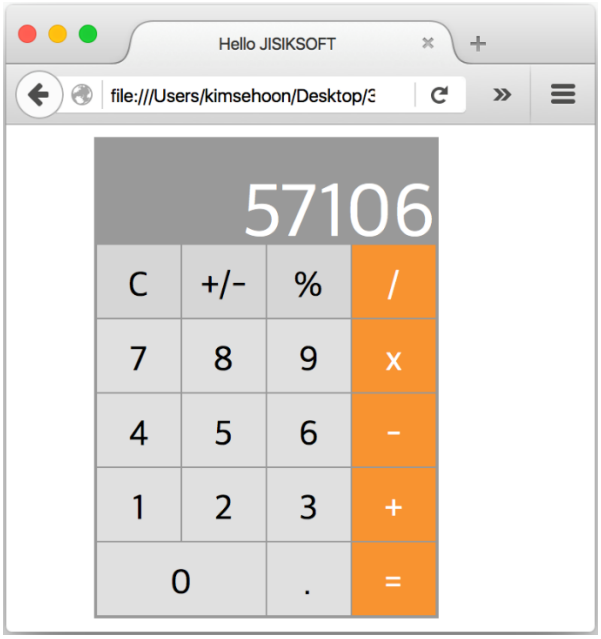
⑰ 계산된 결과는 firstOperand 변수에 저장되는데 소수점의 위치를 계산해서 decimalPlace의 값을 변경해야한다.

⑱ 소수점의 위치를 계산해서 정수면 0의 값을 반환하고, 실수이면 소수점 오른쪽의 숫자의 개수를 카운트Count해서 반환한다. JavaScript의 split() 함수를 사용해서 문자열을 '.'(마침표)를 기준으로 나누어주는데, 모든 프로그래밍에서 문자열을 가지고 조작하는 방법은 많이 사용한다. split() 함수는 배열을 반환하기 때문에 'num.split('.')[1]'과 같이 두 번째 배열이 소수점 오른쪽의 숫자들을 의미하며 이 숫자들의 갯수를 카운트해서 함수의 결과로 반환한다.

⑲ 계산기의 가장 위에 결과를 보여주는데, 숫자의 갯수가 많으면 크기를 작게 해주어야 한다. 여기서는 숫자가 9개 이상이면 순차적으로 숫자의 크기가 조금씩 작아지게 된다. 또한 숫자가 16개 이상이면 계산을 더이상 하지 않는다.

확인 사이트: http://jisiksoft.com/book/4/chapter_2/2.3/calculator.html

그림 2-5 정상적으로 동작하는 계산기 결과



이번 Chapter에서는 브라우저에서 사용할 수 있는 계산기를 간단히 만들었는데, HTML 언어에서 태그만을 사용한 것을 좀더 주의깊게 볼 필요가 있다. 즉, 모든 디자인은 사각형에 기반해서 만들면 되며, 위치와 크기 등을 조정하기 위해서 픽셀을 변경하는 수작업에 다소 시간이 소요되지만 의외로 깔끔한 디자인들을 만들 수 있는 가장 간단한 방법이다. 이후에 만들어지는 대부분의 웹 프로그램들도 태그 위주로 진행되기 때문에 독자들도 태그를 좀더 친숙하게 받아들였으면 한다. "1.2 CSS" 장에서 태그의 위치를 이해하는 것으로 예제를 만든 것은 이러한 이유 때문이다.

chapter 3

설문조사(Survey) 만들기

영화관 매표소에서 영화 티켓을 끊고나서 경품을 주는 설문조사에 참여한 적이 있었는데, 인터넷 세상에서는 설문조사도 핸드폰으로 할수 있게 만들었다. 많은 다양한 매장에서도 참여자 전원에게 소정의 상품을 주면서 설문조사에 참여하는 이벤트를 진행하곤 하는데, 이번 Chapter에서 만드는 것은 작은 화장품 매장에서 진행할 수 있는 설문조사를 만든다. 이를 이용해서 독자들도 매장의 객관적인 평가를 받아볼 수도 있을 것이다. 참고로 문자를 받은 고객의 10% 미만이 설문조사에 참여한다고 한다. 또한, 우리들은 스팸문자와 스팸메일의 홍수 속에서 살기 때문에 설문조사는 1년에 한번만 진행하는 것이 좋다고 한다. 이제부터 만드는 웹 페이지는 화장품 매장의 설문조사인데, 매장에 방문했던 고객들은 문자를 받은 후 핸드폰이나 브라우저에서 설문조사에 참여한다. 또한 설문을 진행하는 관리자는 설문조사의 결과를 막대그래프로 보고 엑셀로 다운받을 수 있다.

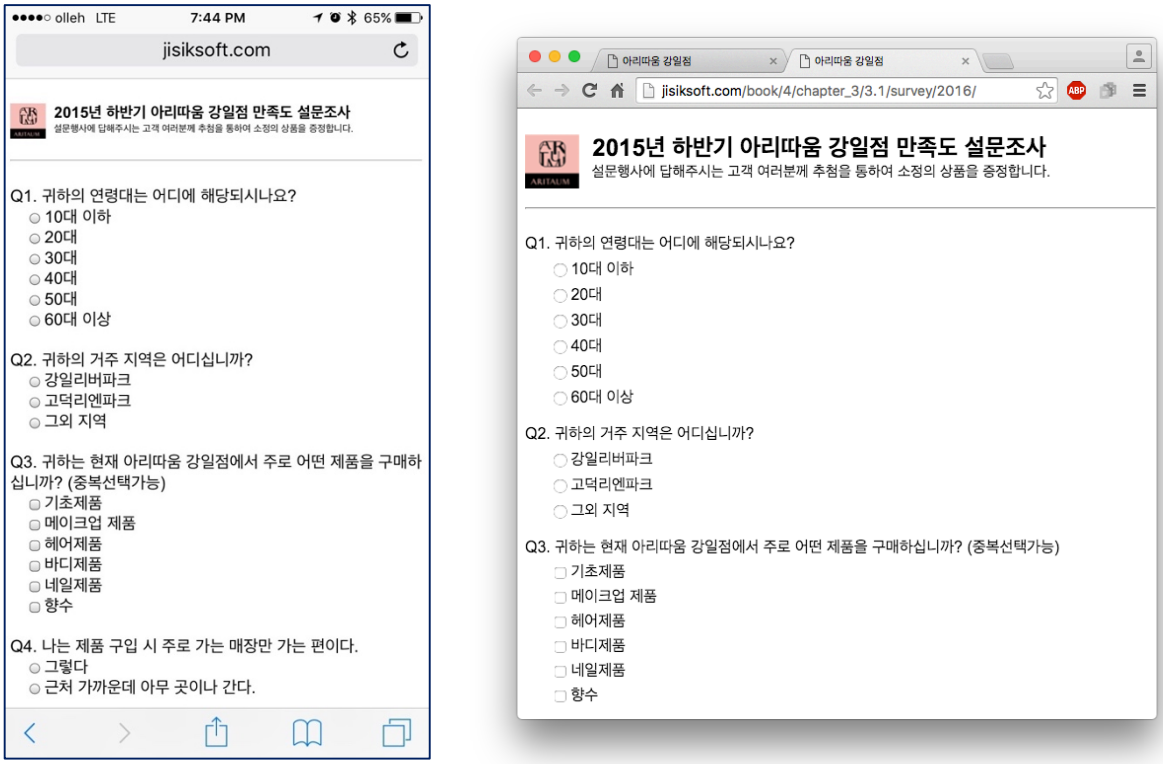
3.1 설문조사 디자인

[그림 3-1]은 설문조사의 화면을 보여주는데, 고객이 선택해야 하는 입력 항목들이 상당히 많다. 보기의 항목 중에서 한 가지만을 선택할 때는 라디오Radio 입력Input으로 만들어주고, 다수의 항목을 중복선택이 가능하게 하는 것은 체크박스CheckBox를 사용한다. 그래서 [코드 3-1]과 같이 고객의 입력을 받는 항목은 <input></input> 태그를 사용했으며, 입력받는 형식을 의미하는 type의 값을 'radio' 또는 'checkbox'로 설정하였다. <input> 태그에는 디자인을 위하여 클래스 이름을 사용하고, 질문 항목에 접근하기 위해서 같은 질문의 태그들은 같은 이름Name을 갖는다. 또한 질문의 항목이 몇 번째인지를 확인하기 위해서 <input> 태그의 값은 선택항목의 숫자를 갖는다. 항목을 선택하는 질문에는 하나의 질문에 다수의 <input> 태그가 사용되었기 때문에 id 값을 사용하지 않고 이름Name 값을

사용했음을 주의깊게 봐야하며, 'Q10'과 'Q11'은 고객이 의견을 작성해야 하기 때문에 하나의 <TEXTAREA> 태그를 사용하고 태그의 내용에 접근하기 위해서 id 값을 사용했다. [코드 3-2]는 설문조사의 디자인을 위한 CSS 파일의 내용이며, 설문조사의 디자인은 통일된 것이 많기 때문에 입력된 내용이 많아도 CSS 디자인은 간결하게 만들 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_3/3.1/survey/2016/

그림 3-1 고객이 보게 되는 설문조사의 해드폰과 브라우저 화면



[코드 3-1]에서 <input> 태그에 name 속성으로 질문 항목을 구분하는 것은 예전부터 많이 사용된 방법이며, 이와 같이 사용자가 선택을 하는 항목에서 name 속성이 가장 많이 사용된다는 것을 이해하면 된다. 필자가 그동안 웹 프로그래밍을 하면서 name을 사용한 경우는 설문조사를 만들 때가 대부분이었고 다른 프로그램에서는 id 속성만을 주로 사용하였다. 1.2장의 [그림 1-25]에서 class, name, id 속성의 차이점에 대해서 간략히 설명했었는데, 여기서는 name의 사용을 눈여겨볼 필요가 있다. 유일한 값을 가지는 id 속성과 달리 class와 name 속성은 중복이 가능하지만, 대부분은 class 속성을 사용하고 설문조사와 같은 다중 선택 항목에서만 name 속성을 사용한다.

[코드 3-1] 설문 조사 HTML (/2015/index.html)

```
<!DOCTYPE html>

<head>
    <title>아리따움 강일점</title>
    <meta charset="utf-8"></meta>
    <link rel="stylesheet" href="./css/survey.css"></link>
</head>
<body>
    <br>
    <div>
        <div id="logo">                                                    //---①
            <img src="./img/logo_aritaum.png">
        </div>
        <div>
            <div class="bold font_24">                                     //---②
                  2015년 하반기 아리따움 강일점 만족도 설문조사
            </div>
            <div class="font_15">
                  설문행사에 답해주시는 고객 여러분께 추첨을 통하여
                        소정의 상품을 증정합니다.
            </div>
        </div>
    </div><br>

    <hr></hr>
    <br>

     <div>Q1. 귀하의 연령대는 어디에 해당되시나요?</div>                    //---③
     <input type="radio" class="select" name="q1" value="1"> 10대 이하</input><br>   //---④
     <input type="radio" class="select" name="q1" value="2"> 20대</input><br>
     <input type="radio" class="select" name="q1" value="3"> 30대</input><br>
     <input type="radio" class="select" name="q1" value="4"> 40대</input><br>
     <input type="radio" class="select" name="q1" value="5"> 50대</input><br>
     <input type="radio" class="select" name="q1" value="6"> 60대 이상</input><br><br>

     <div>Q2. 귀하의 거주 지역은 어디십니까?</div>
     <input type="radio" class="select" name="q2" value="1"> 강일리버파크</input><br>
     <input type="radio" class="select" name="q2" value="2"> 고덕리엔파크</input><br>
     <input type="radio" class="select" name="q2" value="3"> 그외 지역</input><br><br>
```

```
<div>Q3. 귀하는 현재 아리따움 강일점에서 주로 어떤 제품을 구매하십니까?
    (중복선택가능)</div>
<input type="checkbox" class="select" name="q3" value="1"> 기초제품</input><br>
<input type="checkbox" class="select" name="q3" value="2">메이크업 제품</input><br>
<input type="checkbox" class="select" name="q3" value="3"> 헤어제품</input><br>
<input type="checkbox" class="select" name="q3" value="4"> 바디제품</input><br>
<input type="checkbox" class="select" name="q3" value="5"> 네일제품</input><br>
<input type="checkbox" class="select" name="q3" value="6"> 향수</input><br><br>

<div>Q4. 나는 제품 구입 시 주로 가는 매장만 가는 편이다.</div>
<input type="radio" class="select" name="q4" value="1"> 그렇다</input><br>
<input type="radio" class="select" name="q4" value="2">
                        근처 가까운데 아무 곳이나 간다.</input><br><br>

<div>Q5. 아모레퍼시픽 제품을 구매할 때 가장 많이 이용하는 곳은 어디인가요?</div>
<input type="radio" class="select" name="q5" value="1"> 아리따움 매장</input><br>
<input type="radio" class="select" name="q5" value="2"> 홈쇼핑</input><br>
<input type="radio" class="select" name="q5" value="3"> 인터넷 쇼핑몰</input><br>
<input type="radio" class="select" name="q5" value="4">
                                    이마트, 롯데마트, 백화점</input><br>
<input type="radio" class="select" name="q5" value="5"> 방문판매</input><br>
<input type="radio" class="select" name="q5" value="6">
                                    종합화장품 가게</input><br><br>

<div>Q6. 제품 구매 시 가장 영향을 미치는 요인은 무엇인가요? (중복선택 가능)</div>
<input type="checkbox" class="select" name="q6" value="1"> 지인 권유</input><br>
<input type="checkbox" class="select" name="q6" value="2"> 샘플지급</input><br>
<input type="checkbox" class="select" name="q6" value="3"> 판매사원</input><br>
<input type="checkbox" class="select" name="q6" value="4">
                                        브랜드 인지도</input><br>
<input type="checkbox" class="select" name="q6" value="5"> 사용후기</input><br>
<input type="checkbox" class="select" name="q6" value="6"> 사용 경험</input><br>
<input type="checkbox" class="select" name="q6" value="7">
                                    할인기간(멤버쉽 데이)</input><br><br>

<div>Q7. 기능성 화장품 중에서 구매를 원하거나 선호하는 제품은 무엇인가요?</div>
<input type="radio" class="select" name="q7" value="1"> 미백 화장품</input><br>
```

```
<input type="radio" class="select" name="q7" value="2"> 주름개선 화장품</input><br>
<input type="radio" class="select" name="q7" value="3"> 자외선 차단제</input><br>
<input type="radio" class="select" name="q7" value="4">
                                                주름+미백 멀티제품</input><br><br>

<div>Q8. 아리따움 강일점 매장의 청결도 만족도는 어느 정도 인가요?</div>
<input type="radio" class="select" name="q8" value="1"> 매우 만족</input><br>
<input type="radio" class="select" name="q8" value="2"> 대체로 만족</input><br>
<input type="radio" class="select" name="q8" value="3"> 보통</input><br>
<input type="radio" class="select" name="q8" value="4"> 불만족</input><br>
<input type="radio" class="select" name="q8" value="5"> 매우 불만족</input><br><br>

<div>Q9. 아리따움 강일점 직원의 응대 태도는 어떤가요?</div>
<input type="radio" class="select" name="q9" value="1"> 매우 만족</input><br>
<input type="radio" class="select" name="q9" value="2"> 대체로 만족</input><br>
<input type="radio" class="select" name="q9" value="3"> 보통</input><br>
<input type="radio" class="select" name="q9" value="4"> 불만족</input><br>
<input type="radio" class="select" name="q9" value="5"> 매우 불만족</input><br><br>

<div>Q10. 아리따움 강일점 외 타 아리따움 매장을 방문하신다면
          그 이유가 무엇인지요?<br>
                  
   Ex) 제품이 없어서, 제품 카운셀링이 부족해서, 직원의 불친절, 샘플 등</div>
<TEXTAREA class="contents" id="q10" cols=50 rows=4></TEXTAREA>      //---⑤
<br><br>

<div>Q11. 아리따움 강일점에 바라는 점이 있으면 자세히 적어주세요.<br>
        
 적극적으로 개선노력 하겠습니다.</div>
<TEXTAREA class="contents" id="q11" cols=50 rows=4></TEXTAREA>
<br><br><br>

<div class="bold left"> 이     름 :  </div>
<input type="text" class="personal" id="myname"></input><br>                //---⑥
<br>

<div class="bold left"> 핸 드 폰 :  </div>
   <input type="text" class="personal" id="phone"></input><br>                //---⑦
<br>
```

```
    <div id="btn_register">등 록</div><br>                                    //---⑧
      <br><br><br><br><br>

</body>
</html>
```

① 이미지를 넣을 때 <img> 태그를 사용하는데, 필자는 대부분 태그를 사용해서 위치와 크기를 정한다. 그리고, 태그 안에 <img> 태그를 넣어주면 되는데, <img> 태그의 속성은 CSS 파일에서 "width=100%", "height=100%"로 항상 설정해서 태그의 크기에 꽉 차게 만든다.

② 클래스 값으로 'bold'와 'font_24'와 같이 두 개의 개별 값들을 중복으로 사용해서 디자인 속성의 내용을 유추할 수 있게 만들 수 있다. CSS 파일에서 'bold'는 굵은 글씨로 정의되었고, 'font_24'는 글자 크기가 24 픽셀을 의미한다.

③ 질문은 태그를 사용했으며, 여기서는 디자인 속성을 사용하지 않았다. 이와 같은 경우에는 <body>에서 부여된 속성을 자동적으로 상속받게 된다.

④ 고객이 선택하는 항목들은 <input> 태그를 사용했는데, 클래스 값은 디자인을 위해서 사용했다. 또한 질문의 항목들을 구분하기 위해서 name 속성을 사용했으며, 여기서는 첫 번째 질문에 대한 선택항목으로 'q1'(Question 1)이라는 값을 갖는다.

⑤ 여러 줄의 장문을 입력하기 위해서는 <TEXTAREA> 태그를 사용하는데, rows 속성은 보여지는 세로줄 수를 의미하고 cols는 가로로 쓸 수 있는 글자의 갯수를 의미한다. 여기서는 유일한 항목이기 때문에 id 속성을 사용했다.

⑥ 고객의 이름을 받는 항목은 텍스트로 받기 때문에 <input> 태그의 형식을 의미하는 type 속성이 "text"라는 값을 갖는다. 클래스 이름은 디자인을 위해서 사용하며, id 값은 이후에 고객의 이름에 접근하기 위해서 사용한다.

⑦ 고객의 전화번호를 받는 항목도 텍스트로 입력받으며, 이후에 전화번호에 접근하기 위해서 id 값으로 'phone'을 사용했다. 이와 같이, id 값은 일반적인 프로그램에서 변수 값을 정할 때와 같이 값에서 특징을 유추할 수 있게 명명해야 한다.

⑧ 지금까지 버튼은 태그를 사용했으며, 여기서도 "btn_register"라는 id 값을 가지는 태그 버튼을 만들었다.

[코드 3-2] 설문 조사 CSS (/2015/css/survey.css)

```
body {                                                                    //---①
    width:640px;
    font-family:arial;
```

```
        font-size:16px;
}
#logo {
        width:55px;
        height:55px;
        float:left;
}
img {                                                          //---②
        width:100%;
        height:100%;
}
.select, .contents {                                           //---③
        margin: 10px 0px 0px 30px;
}
.contents, .personal {
        font-size:16px;
        width:450px;
}
.personal {
        position:relative;
        top:-4px;
        width:150px;
}
#btn_register {                                                //---④
        position:relative;
        top:20px;
        left:200px;
        width:130px;
        height:50px;
        border-radius: 25px;
        color:#ffffff;
        font-size:20px;
        line-height:50px;
        text-align:center;
        background-color:#1646a7;
        cursor:pointer;
}
.left { float:left; }                                          //---⑤
.under { text-decoration: underline; }
.arial { font-family:arial; }
.bold { font-weight:bold; }
.font_15 { font-size:15px; }
.font_24 { font-size:24px; }
```

① body 태그는 브라우저에서 보여지는 내용의 전체를 의미하기 때문에 여기서 설정된 디자인 속성은 모든 태그들에 공통적으로 적용된다.

② 이 책에서 사용하는 모든 <img> 태그들은 가로와 세로 값은 모두 100%로 설정되며, <img> 태그를 감싸고 있는 태그를 만들어서 위치와 크기를 정

한다.

③ CSS 디자인 속성에서 다수의 클래스나 id에 공통적으로 적용되는 디자인은 ','(콤마)를 사용해서 공통의 디자인을 만든다.

④ '등록' 버튼의 디자인을 정했는데, 위치Position는 relative로 설정했다. 1.2장의 CSS에서 태그의 위치에 비중을 두고 디자인을 설명한 이유는 위치에 대하여 정확히 이해하는 것이 필자의 경우에 가장 어렵고 중요했다. 다른 디자인 속성들은 인터넷에서 쉽게 찾고 이해하기 쉽지만, 디자인의 가장 중요한 것은 사각형의 위치와 크기라고 필자는 생각한다.

⑤ 클래스에 다수의 이름을 부여해서 디자인을 쉽게 적용하기 위해서 여기서도 하나의 속성값을 가지는 간단한 클래스 CSS 디자인을 만들었다.

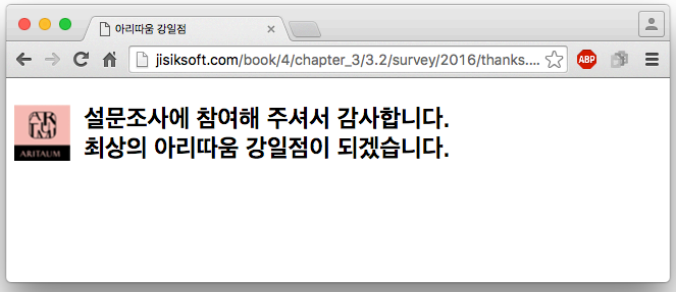
CSS 디자인을 적용할 때는 처음에 많은 시간이 소요되는데, 디자인 속성을 넣거나 빼고, 또한 값을 변경하면서 브라우저에서 확인해야 하기 때문에 웹 프로그래밍을 할 때는 충분한 여유를 가지고 진행하는 것이 좋다. 필자는 픽셀 값을 조금씩 조정하느라 몇 시간을 보낸 적도 있다. 지금까지 설문조사 화면의 내용과 디자인을 만들었으며 다음장에서는 고객이 등록버튼을 클릭했을 때 동작하는 이벤트를 처리하는 JavaScript 코드를 구현한다.

3.2 설문조사 이벤트

설문조사 이벤트는 '등록' 버튼이 클릭되었을 때 발생되며, JavaScript 함수가 실행되어서 선택되지 않은 문항이 있는지를 확인하고 서버로 결과를 보낸다. 또한 결과가 정상적으로 등록되면 [그림 3-2]과 같은 결과 페이지를 보게 된다. 설문조사를 만드는 프로그래머는 서버에서 정상적으로 저장되었을 때 결과 페이지가 브라우저에서 보이게 만들면 된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_3/3.2/survey/2016/

그림 3-2 설문조사가 정상적으로 이루어진 이후의 결과 페이지



서버에 저장되는 내용은 브라우저에서 만들어지는데, [그림 3-3]은 저장하기 위하여 브라우저에서 만드는 데이터를 보여준다. JSON 데이터를 사용하지 않고 이와같이 문자열로 만들어서 사용하면 되는데, 파일에 저장하기 위해서는 문자열로 만들어서 사용하는 것이 효과적이다. 문자열로 데이터를 저장하기 위해서는 데이터를 구분할 수 있는 특수문자를 사용하는게 일반적이다. 여기서는 고객을 구분하기 위해서 '&&&&'를 사용했고, 한 고객의 데이터들을 구분하기 위해서 '##'를 사용했다. 또한, 체크박스Checkbox 항목은 중복된 선택이 가능한데, 이것을 위해서 '&' 문자를 사용했다. 질문에 대한 답은 일반적으로 숫자와 문자로 이루어지기 때문에 이와 같이 잘 사용하지 않는 특수문자들을 사용해서 문자열 데이터를 만드는 방법은 실제로 많이 사용한다. 문자열을 만드는 것은 문자열들을 더하면 되기 때문에 JavaScript에서는 쉬운 방법이며, 또한 특정 문자열을 기준으로 분리하는 것도 간단하다. 이번 장에서는 특정문자들을 가지고 문자열을 만드는 방법이 사용되며 이렇게 만들어진 문자열을 서버로 보내고 서버에서는 파일에 저장하기만 하면 된다. 문자열 안에서 다양한 데이터들을 넣는 위치는 프로그래머가 정하면 된다. 여기서는 고객의 이름과 전화번호를 앞쪽에 위치하게 만들었다.

그림 3-3 서버에 저장되는 설문조사 참여자의 데이터 구조

<저장되는 데이터>

```
&&&&남혜정##010-1234-5678##2##3##2&5##2##1##2&7##4##1##1##
급하게사야할때 ##제품종류가더많았으면좋겠어요##20160104
```

<데이터 분석>

```
&&&&                                    1. 고객 구분 문자
남혜정                                  2. 이름
##
010-1234-5678                           3. 전화번호
##2##3##2&5##2##1##2&7##4##1##1##       4. 문제 (q1~q9)
급하게사야할때                          5. 문제 (q10)
##
제품종류가더많았으면좋겠어요           6. 문제 (q11)
##
20160104                                7. 날짜
```

[코드 3-3]은 설문조사의 '등록' 버튼이 클릭되었을 때 처리되는 JavaScript 함수들을 보여주는데, 설문조사에서 중요한 것은 답을 주지 않은 문항이 있는지를 확인해야 한다. 서버와의 통신에서 중요한 것은 브라우저에서 처리할 수 있는 부분은 될 수 있으면 서버에서 처리하지 않는 것이 좋다. 아주 많은 사용자가 접속하는 서버를 다루어 본 필자는 서버에서 해야할 일을 최대한 단순화시키려 노력하는 편이다. 예를 들어, 보안이 중요한 요즘에는 서버에서 암호화를 진행해야 하는 경우가 많은데, 암호화하는 과정조차도 서버에게는 상당한 업무가 될 수 있으므로 사용자가 많이 접속하는 서버에서는 서버에서 처리할 수 있는 한계를 높이는 것이 시간과 비용면에서 상당히 중요하다. 이번 장에서 보여주는 JavaScript 코드는 일반적인 사이트에서 예전부터 많이 사용하는 방법이라 생각하고 코드를 충분히 이해하기를 바란다.

[코드 3-3] 설문조사의 '등록' 버튼이 클릭되었을 때 실행되는 JavaScript 함수 (/js/survey.js)

```
function checkRadio(objRadio){                                    //---①
    var num = objRadio.length;                                    //---②

    for (var i=0; i<num; i++) {
        if(objRadio[i].checked == true){                          //---③
            return objRadio[i].value;
        }
    }
    return 0;                                                     //---④
}

function checkCheckBox(objCheck){                                 //---⑤
    var str = '';
    var flag = false;
    var num = objCheck.length;

    for (var i=0; i<num; i++) {
        if(objCheck[i].checked == true) {
            if (flag) {                                           //---⑥
                str += '&' + objCheck[i].value;
            } else {
                str += objCheck[i].value;
            }
            flag = true;
        }
    }

    if (flag) {
```

```
        return str;
    } else {
        return 0;
    }
}

function register(){                                                    //---⑦
    var str, value;

    if((value = checkRadio($("input[name='q1']"))) != 0) {              //---⑧
        str = '##' + value;
    } else {
        alert("Q1의 질문에 답해주시기 바랍니다."); return false;
    }
    if((value = checkRadio($("input[name='q2']"))) != 0) {
        str += '##' + value;
    } else {
        alert("Q2의 질문에 답해주시기 바랍니다."); return false;
    }
    if((value = checkCheckBox($("input[name='q3']"))) != 0) {           //---⑨
        str += '##' + value;
    } else {
        alert("Q3의 질문에 답해주시기 바랍니다."); return false;
    }
    if((value = checkRadio($("input[name='q4']"))) != 0) {
        str += '##' + value;
    } else {
        alert("Q4의 질문에 답해주시기 바랍니다."); return false;
    }
    if((value = checkRadio($("input[name='q5']"))) != 0) {
        str += '##' + value;
    } else {
        alert("Q5의 질문에 답해주시기 바랍니다."); return false;
    }
    if((value = checkCheckBox($("input[name='q6']"))) != 0) {
        str += '##' + value;
    } else {
        alert("Q6의 질문에 답해주시기 바랍니다."); return false;
    }
    if((value = checkRadio($("input[name='q7']"))) != 0) {
        str += '##' + value;
    } else {
        alert("Q7의 질문에 답해주시기 바랍니다."); return false;
    }
    if((value = checkRadio($("input[name='q8']"))) != 0) {
        str += '##' + value;
```

```
        } else {
            alert("Q8의 질문에 답해주시기 바랍니다."); return false;
        }
        if((value = checkRadio($("input[name='q9']"))) != 0) {
            str += '##' + value;
        } else {
            alert("Q9의 질문에 답해주시기 바랍니다."); return false;
        }
        str += '##' + $("#q10").val() + '##' + $("#q11").val();                    //---⑩

        var str2;

        if(value = $("#myname").val()) {                                           //---⑪
            str2 = '&&&&' + value;
        } else {
            alert("이름을 기입해 주시기 바랍니다.."); return false;
        }
        if(value = $("#phone").val()) {                                            //---⑫
            str2 += '##' + value;
        } else {
            alert("핸드폰 번호를 기입해 주시기 바랍니다.."); return false;
        }
        str2 += str;

        var param = { data : str2 };                                               //---⑬
        var data = doAjax('./ajax_save_survey.php', param);

      if (data == '1') {                                                           //---⑭
        location.replace("./thanks.html");
      }
}

function doAjax(str_url, input_data) {                                             //---⑮
        var result;

        $.ajax({
                    url: str_url,
                    type: 'post',
                    async: false,
                    datatype: 'json',
                    data: input_data,
                    error: function() {
                                alert('등록이 안 되었습니다. 다시 등록해 주시기 바랍니다.');
                    },
                    success: function(obj) {
```

```
                result = obj;
            }
        });
        return result;
}
```

① 질문 항목에서 라디오 <input> 태그의 객체들을 매개변수로 받는데, objRadio 매개변수는 배열Array 안에 모든 라디오 태그 객체가 저장되었다. 'checkRadio()' 함수는 라디오 태그에서 선택된 항목의 값을 결과값으로 전달하며, 선택된 항목이 없으면 0을 return한다. 이 함수를 호출한 곳에서는 0을 받으면 선택된 것이 없기 때문에 고객에게 선택된 항목이 없음을 알려줄 수 있다.

② 배열에 몇 개의 객체들이 저장되었는지를 num 변수에 저장하고, for loop을 객체의 갯수만큰 반복해서 배열의 모든 객체의 선택 유무를 확인한다.

③ 선택된 객체가 있으면 'checked' 속성이 true이기 때문에, 해당 객체의 값(value)을 return한다.

④ 선택된 객체가 없으면 0을 return해서 해당 질문의 라디오 버튼이 선택되지 않았음을 알수 있다.

⑤ 질문 항목에서 체크박스 <input> 태그의 객체들을 매개변수로 받는데, objCheck 매개변수는 배열 안의 모든 체크박스 태그 객체가 저장되었다. 'checkCheckBox()' 함수는 체크박스 태그에서 선택된 항목들의 값들을 '&'를 중간의 구분자로 사용해서 문자열을 만들어서 함수의 결과값으로 전달한다. 만약 선택된 항목이 없으면 0을 return해서, 고객에게 선택된 항목이 없음을 알려줄 수 있다.

⑥ 다수의 체크박스 중에서 선택된 것이 있으면 flag 값이 true로 설정되는데, 가장 먼저 선택된 것은 '&'를 앞에 넣어서 문자열에 붙일 필요가 없지만, 두 번째부터 선택된 항목은 '&'를 사용해서 문자열에 더해주어야 한다. 다수의 항목이 선택되었을 때 중간에 '&' 문자를 넣기 위해서 flag 변수를 사용했으며, 이와 같은 방법은 문자열을 다루는 많은 프로그래밍에서 사용한다.

⑦ '등록' 버튼이 클릭되었을 때 register() 함수가 실행되며, 질문에 답한 내용이 없는지를 확인하면서 문자열을 만들고 서버로 결과를 전송한다.

⑧ 첫 번째 질문의 라디오 <input> 태그 객체들은 name을 'q1'으로 갖는데, jQuery를 사용해서 $("input[name='q1']")과 같이 하게 되면 'q1'을 name으로 갖는 모든 <input> 태그들의 객체를 배열에 저장하게 된다. checkRadio() 함수를 실행할 때 jQuery를 사용해서 매개변수를 전달해주고 결과값을 value 변수에 저장한다. 그런데, 만약 value 변수의 값이 0이면 선택된 항목이 없기 때문에 경고 창을 출력해서 질

문에 답을 요구하고, 0이 아니면 선택된 항목의 값을 문자열에 더해준다.

⑨ 세 번째 질문은 체크박스 <input> 태그 객체이고 name을 'q3'으로 갖는데, jQuery를 사용해서 $("input[name='q3']")과 같이 하게 되면 'q1'을 name으로 갖는 모든 <input> 태그들의 객체를 배열에 저장하게 된다. checkCheckbox() 함수를 실행하고 결과값을 문자열에 더해주는데, 동작원리는 ⑧에서 설명한 내용과 같다.

⑩ 고객의 의견을 듣는 항목인 열 번째와 열한 번째의 질문은 빈 값이어도 괜찮기 때문에, 해당 내용의 값이 있는지를 확인하지 않고 문자열에 더해주었다.

⑪ 고객의 이름은 결과 문자열에서 맨 앞에 위치하며, 고객 간의 데이터를 구분하기 위해서 구분자로 '&&&&'를 사용하였다. 고객의 이름도 값이 없으면 이름을 기입하는 경고 창을 화면에 보여주어야 하며, 이름은 value 변수에 저장되는데 해당 항목에 이름이 없으면 value 변수는 null의 값을 갖기 때문에 경고 창이 출력된다.

⑫ 핸드폰 번호의 값은 value 변수에 저장되고 문자열에 더해지게 된다. 이때, 핸드폰 번호가 입력되지 않았으면 value 변수는 값이 없다는 의미의 null 값을 갖게되고 경고 창을 출력한다.

⑬ 모든 질문에 대한 답이 이루어지고 문자열이 만들어졌으면 해당 문자열을 서버로 보내게 되는데, Ajax를 사용해서 서버의 './ajax_save_survey.php' 함수를 호출한다.

⑭ 데이터가 서버에 정상적으로 저장되면 'thank.html'이라는 결과 페이지를 브라우저에서 여는데, JavaScript의 location.replace() 함수를 사용해서 현재의 창에서 결과 페이지를 열게 된다.

⑮ 이 책에서 서버와의 통신을 위해서 사용하는 방법은 jQuery의 $.ajax() 함수를 사용하는데, doAjax() 함수를 만들어서 모든 서버와의 통신에서 공통적으로 사용한다. 이 함수에 대한 설명은 1.4장의 PHP에서 자세히 다루었다.

[코드 3-4]는 서버에서 실행되는 PHP 파일이며, 'Post 방식'으로 전달받은 데이터를 해당 파일에 저장한다. 여기서 사용되는 파일은 3개인데 설문에 참여한 고객의 숫자를 저장하기 위해서 'count.data' 파일이 사용되며, 모든 고객의 데이터를 저장하기 위해서 'alldata.data' 파일이 사용된다. 또한 해당 날짜를 이름으로 가지는 파일(ex: '20160104.data' : 2016년 1월 4일)을 만들어서 참여한 날의 데이터들만 따로 저장한다. 파일 입출력에 대한 내용은 필자의 두 번째 책인 "주식 분석 프로그램 만들기"에서 자세히 설명했는데, 파일을 열어서 메모리에 올리고 내용을 읽거나 쓴 이후에 마지막으로 파일을 닫으면 된다. 기본적인 파일 입출력에 대하여 이해를 하고 있

으면 모든 프로그래밍 언어가 그에 따른 예약어를 만들어서 사용하는 것을 쉽게 알 수 있다. PHP 언어에서는 파일을 열기 위해서 fopen() 함수를 사용하고, 읽거나 쓰기 위해서 fread()와 fwrite() 함수를 사용하며, 파일을 닫기 위해서 fclose() 함수를 사용한다.

[코드 3-4] 설문조사 데이터를 서버에 저장하기 위한 PHP 함수 (ajax_save_survey.php)

```
<?php

$data;
if(isset($_POST['data'])) { $data = $_POST['data']; }                    //---①

$date = date('Y').date('m').date('d');                                   //---②
$data .= "##".$date;                                                     //---③

$filename = "./data/".$date.".data";                                     //---④
$handle = fopen($filename, "a+");                                        //---⑤
fwrite($handle, $data."\n");
fclose($handle);

$filename = "./data/alldata.data";                                       //---⑥
$handle = fopen($filename, "a+");
fwrite($handle, $data."\n");
fclose($handle);

$count;

$filename = "./data/count.data";                                         //---⑦
$handle = fopen($filename, "r");
$count = fread($handle, filesize($filename));
fclose($handle);

$count += 1;                                                             //---⑧

$filename = "./data/count.data";                                         //---⑨
$handle = fopen($filename, "w");
fwrite($handle, $count);
fclose($handle);

print 1;                                                                 //---⑩
exit();
```

```
?>
```

① 'Post 방식'의 데이터를 서버에서 받을때 1.4장에서는 "A ? B : C" 연산자를 사용했지만, 여기서는 if 연산자를 사용해서 고객의 브라우저에서 보낸 데이터를 $data 변수에 저장한다.

② $date 변수는 현재의 날짜를 갖게 되는데, PHP의 예약함수인 data() 함수를 사용해서 '년도/월/일'의 값들을 개별적으로 얻어서 문자열로 저장한다.

③ 고객으로부터 받은 데이터는 날짜가 없기 때문에, 현재의 날짜를 저장하고 있는 $date 값을 문자열에 더해준다. 이렇게 해서 저장되는 데이터가 완성된다.

④ 현재의 날짜를 가진 파일이름을 정하고 데이터를 넣게되는데, 예를 들면, '20160104.data'와 같은 이름을 가진 파일에 고객의 데이터를 저장한다.

⑤ 파일을 열고, 파일의 내용 뒤에 현재의 고객 데이터를 추가하기 위해서 "a+" 속성을 가지고 fopen() 함수를 실행한다. $handle 변수는 메모리에 올라온 파일의 주소 값을 가지고 있다고 이해하면 되는데, 이와 같이 파일을 열게 되면 메모리에 파일의 데이터가 올라가고 변수를 통해 접근이 가능하게 된다. fwirte() 함수는 파일에 데이터를 쓰게 되는데, 파일을 열때 "a+" 속성을 사용했기 때문에 파일의 마지막에 $data 변수의 내용이 추가된다. 파일을 닫는 함수인 fclose()를 사용해서 메모리에 올라온 내용을 서버의 디스트에 저장한다.

⑥ 모든 고객의 데이터는 'alldata.data' 파일에 저장되는데, ⑤에서와 같은 방법으로 데이터를 추가한다.

⑦ 설문조사에 참여한 고객의 숫자는 'count.data' 파일에 저장되는데, 파일의 숫자를 읽고 값을 증가시킨 후에 파일의 내용을 변경하면 된다. 여기서는 "r"을 사용해서 읽기 모드로 파일을 열고 값을 $count 변수에 저장한다.

⑧ 고객의 데이터가 하나 추가되기 때문에 설문조사에 참여한 고객의 수도 1 증가시키기 위해서 $count 변수에 1을 더한다.

⑨ 설문조사에 참여한 고객의 수가 변경되었기 때문에 'count.data'의 값도 변경되어야 한다. 그래서, 'count.data' 파일을 쓰기 모드('w')로 열어서 값을 덮어쓴다.

⑩ 서버의 모든 동작이 에러없이 진행되었기 때문에 1을 브라우저에 반환하고, 고객의 브라우저에서는 성공적으로 설문이 등록되었음을 인지하고 결과 페이지를 열게 된다.

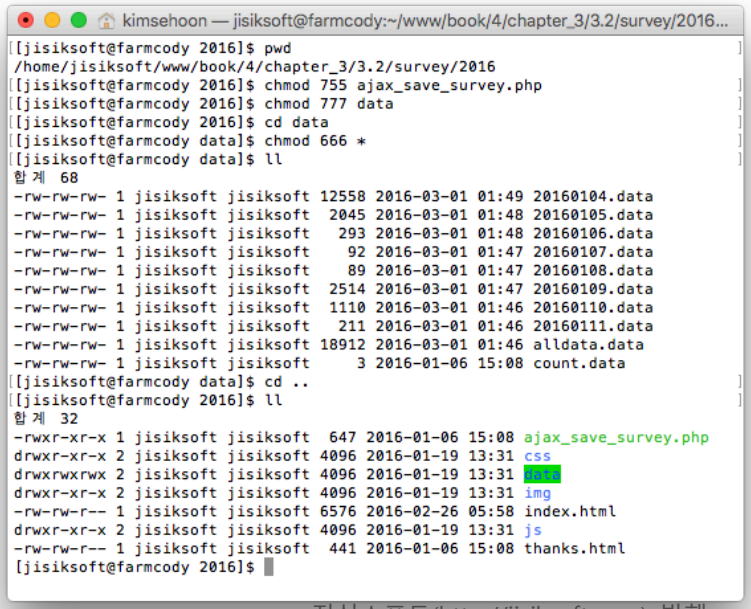
Linux 서버에서 프로그래밍을 하다보면 파일의 접근권한Permission이 올바르게 설정되지 않아서 프로그램의 실행이 원활하지 않을 때가 종종 발생한다. 인터넷에 연결되

서버는 외부의 사용자와 통신을 하기 때문에 내부와 외부의 사용자들이 접근할 수 있는 방법들을 세분화해 놓았는데, 여기서 사용하는 PHP 파일과 데이터 파일들은 외부에서 접속한 사용자가 실행하는 의미이기 때문에 파일의 접근권한을 변경해 줄 필요가 있다. 파일의 접근권한을 변경하는 Linux 명령어는 'chmod'(Change Mode)이며 인터넷에 사용방법이 자세히 설명되어 있고, 여기서 chmod 명령어를 설명하면 주제와 벗어난 부분에 대해서 너무 많은 설명을 해야하기 때문에 설명을 생략한다. [그림 3-4]는 파일에 데이터를 사용하기 위해서 실행하는 명령어의 사용법을 보여주며, [그림 3-5]는 코드에서 사용한 실제 예제를 보여준다. 간단히 설명하면 브라우저를 통해서 실행되는 'ajax_save_survey.php' 파일은 외부 사용자가 실행할 수 있어야 하고, 데이터 파일이 저장된 '/data/' 디렉토리는 "읽기/쓰기/실행"의 세 가지 모드가 가능하도록 설정되었다. 또한 '/data/' 폴더에 있는 데이터 파일들은 읽고 쓸수 있도록 설정되었다. 우리가 이해해야 할 부분은 서버는 보호되어야 하는데, 브라우저를 통해 파일에 접근하려는 사용자는 외부사용자(기타 사용자 권한)라는 것이다. 파일에 대한 접근권한을 올바르게 설정해야 한다는 것을 PHP와 같은 서버 프로그래밍을 하는 프로그래머는 충분히 이해하고 있어야 한다. 프로그래밍을 하면서 부딪히는 문제 중의 하나이기에 여기서 간단히 설명했다.

그림 3-4 데이터를 저장하기 위한 파일의 접근권한 변경

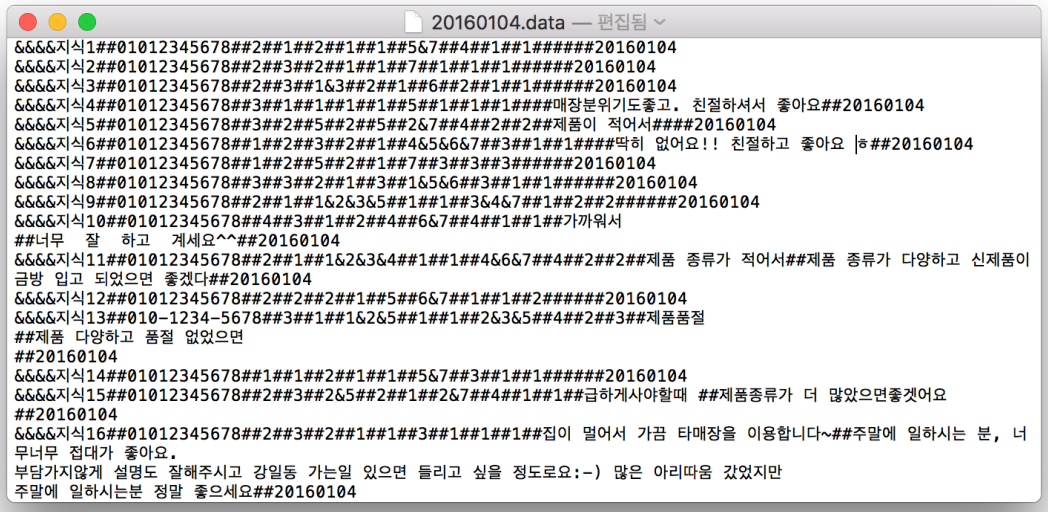
```
chmod 755 ajax_save_survey.php   // 실행파일로 만든다.
chmod 777 data                   // data 디렉토리의 접근이 가능하다.
cd data
chmod 666 *                      //  data 디렉토리의 모든 파일은 읽기/쓰기가 가능하다.
```

그림 3-5 파일의 접근권한을 실행한 예제



[그림 3-6]은 고객의 데이터가 파일에 어떻게 저장되는지를 보여주는데, 브라우저에서 만들어진 문자열을 받아서 마지막에 설문에 참여한 날짜가 추가되어 파일에 저장되었다. 설문조사에서 가장 중요한 것은 고객의 의견이며 데이터에 저장된 내용이 설문조사에서 가장 중요한 정보가 된다.

그림 3-6 고객의 데이터가 저장된 파일의 내용



고객이 보는 설문조사는 생각보다 간단할 수 있다. 고객의 데이터는 데이터베이스에 저장할 수도 있지만, 몇천 명 이상되는 데이터가 아닌 이상은 파일에 저장해도 분석하기 위한 시간이 그다지 길지는 않다. 다음 장에서는 파일에 저장된 내용을 가지고 관리자가 쉽게 볼수 있는 챠트Chart 형식으로 볼 수 있는 페이지를 만든다. 설문조사를 만드는 프로그래머에게 의뢰하는 사람은 매장 주인이기 때문에 매장 주인이 실시간으로 쉽게 고객의 의견을 볼수 있게 만드는 것이 아주 중요하다.

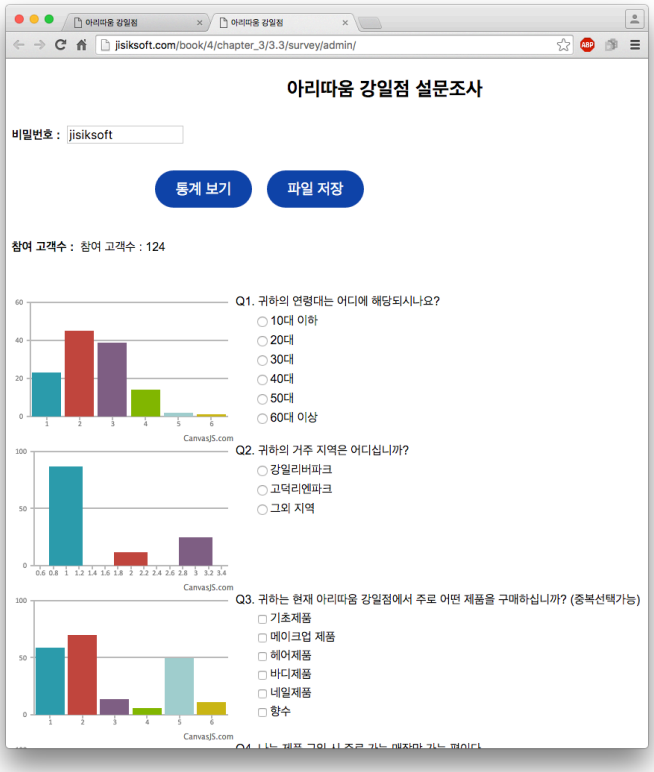
3.3 그래프를 이용한 결과 보기

관리자가 보는 화면은 숫자가 아닌 챠트Chart를 사용해서 결과를 보여주는데, 챠트를 직접 만드는 것도 가능하지만 인터넷을 통해서 챠트를 만들수 있는 라이브러리Library를 사용하는 것이 시간적인 측면에서 좋다. 이 책에서는 간단한 막대그래프를 그려

주기 위해서 라이브러리를 사용하지만, 많은 라이브러리들은 더 많은 다양한 챠트를 제공하기 때문에 막대그래프를 만드는 것을 연습으로 생각하고 독자들은 다양한 챠트로 응용해보기를 권한다. 라이브러리를 만든 회사들은 홍보의 개념으로 개인들에게는 무료로 제공하고, 상업적으로 사용할 때만 비용을 지불하는 정책을 사용한다. 챠트를 만드는 라이브러리들은 예전부터 많이 있었는데 HTML5의 Canvas(그림 그리는 표준 방법)가 발표되기 전에는 HTML의 태그를 추가해서 그래프를 만들어주는 방법을 사용했다면, 여기서 사용하는 'CanvasJS' 챠트는 JavaScript 언어를 사용해서 자동으로 그림을 그려주기 때문에 속도가 기존의 것보다 많이 빠르다. 즉, 기존의 태그를 추가하는 방법은 XML 형식의 내용이 추가되고 브라우저가 해석해서 그래프를 그려주었다면, Canvas는 직접 코드로 그림을 그리기 때문에 빠르다. [그림 3-7]은 설문조사의 결과를 보여주는 관리자 페이지이며, 'CanvasJS' 챠트를 무료로 사용하기 때문에 챠트의 우측 아래에 'CanvasJS.com'이 표시된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_3/3.3/survey/admin/ (비밀번호: 'jisiksoft')

그림 3-7 관리자가 볼 수 있는 설문조사 통계 페이지



CanvasJS 챠트를 사용하기 위해서는 http://canvasjs.com/ 홈페이지에서 라이브러리를 다운받아서 사용하면 되는데, 다양한 종류의 챠트를 선택해서 CanvasJS에서 제공하는 함수를 사용해서 데이터를 가지고 챠트를 만들면 된다. 모든 프로그래밍 언

어에서 라이브러리라고 하면 함수의 집합이라고 생각하면 된다. 클래스로 만들어진 라이브러리도 결국은 클래스 안에 정의된 함수를 사용하는 것이기 때문에 프로그래밍에서는 필요한 함수들을 잘 찾아서 사용하는 것이 아주 중요하다. 특정 라이브러리를 사용할 때는 생각보다 많은 시간이 소요되는 편인데, 사용하는 방법을 이해해야 하고 자신이 원하는 챠트를 만들기 위해서 세세한 부분까지 이해해야 하기 때문이다. 이번 장에서는 기본챠트를 그리기 때문에, 챠트를 그리기 위한 데이터를 JSON 데이터 형식으로 만들어서 삽입하는 간단한 방법을 보여주지만 이것을 이해하면 다른 형식의 챠트에 데이터를 넣는 것도 같기 때문에 쉽게 만들 수 있을 것이다.

[코드 3-5]는 관리자 페이지를 만들기 위한 HTML 코드이며, 고객이 참여하는 설문조사 페이지의 내용에 비밀번호와 챠트를 그려주는 영역이 추가되었다.

[코드 3-6]은 서버로부터 고객의 모든 데이터를 받아서 구분자('&&&&', '##', '&')를 기준으로 내용들을 나누어서 처리하는 JavaScript 함수를 보여주는데, 웹 프로그래밍은 이와 같이 자신이 원하는 것들을 직접 JavaScript 언어로 만드는 것이 중요하다. JSON 데이터로 만들 지, 아니면 설문조사처럼 문자열로 처리할 지는 프로그래머가 정하면 되는데 데이터 구조를 정의하는 것이 프로그래머이기 때문에 어떠한 형태이든 필요에 따라서 적당한 방식을 정하면 된다.

[코드 3-5] 설문조사의 관리자 페이지를 만드는 HTML 코드 (/admin/index.html)

```
<!DOCTYPE html>

<head>
    <title>아리따움 강일점</title>
    <meta charset="utf-8"></meta>
    <link rel="stylesheet" href="./css/admin.css"></link>
    <script src="./js/jquery-2.1.3.min.js"></script>
    <script src="./js/canvasjs.min.js"></script>                    //---①
    <script src="./js/admin.js"></script>
</head>
<body>
     <br>
     <div id="mainTitle"> 아리따움 강일점 설문조사 </div>
     <br/><br>

     <div class="bold left">비밀번호 :  </div>
     <input type="text" class="personal" id="passwd" value=""></input>          //---②
     <br><br><br><br><br><br>
```

```
        <div class="btn" id="btn1" onclick="getAllData()">통계 보기</div><br>          //---③

        <div class="btn" id="btn2" onclick="downloadFile()">파일 저장</div><br>     //---④

        <div class="bold left">참여 고객수 :  </div>
        <div id="count"> - </div>                                                    //---⑤
        <br><br><br>

        <div class="chart" id="chart1"></div>                                        //---⑥
       <div>Q1. 귀하의 연령대는 어디에 해당되시나요?</div>                          //---⑦
        <input type="radio" class="select"> 10대 이하</input><br>
        <input type="radio" class="select"> 20대</input><br>
        <input type="radio" class="select"> 30대</input><br>
        <input type="radio" class="select"> 40대</input><br>
        <input type="radio" class="select"> 50대</input><br>
        <input type="radio" class="select"> 60대 이상</input><br><br>

        <div style="clear:left;"> </div>                                             //---⑧
        <div class="chart" id="chart2"></div>
        <div>Q2. 귀하의 거주 지역은 어디십니까?</div>
        <input type="radio" class="select"> 강일리버파크</input><br>
        <input type="radio" class="select"> 고덕리엔파크</input><br>
        <input type="radio" class="select"> 그외 지역</input><br><br>

        ##### 생략 #####

</div>
</body>
</html>
```

① CanvasJS 라이브러리를 사용하기 위해서는 http://canvasjs.com 사이트에서 'canvasjs.min.js' 파일을 받아서 링크해서 사용하면 된다. 일반적으로 jQuery나 CanvasJS에서 제공하는 파일들의 이름 중간에 '.min.'과 같은 문자가 추가되는 것은 파일의 크기를 줄이기 위해서 압축을 사용한 파일임을 의미한다. JavaScript 언어로 만들어진 파일들은 브라우저에서 코드의 확인이 가능한데, 이렇게 압축된 파일들은 정상동작을 하면서도 코드를 분석하는 것이 어렵다. 파일의 크기를 작게하는 것은 네트워크 트래픽을 줄이는 의미도 있지만, 다른 사용자가 쉽게 코드를 복사해서 사

용하는 것을 막기 위함도 있다.

② 고객의 설문조사 데이터에는 이름과 전화번호가 있기 때문에 비밀번호를 사용해서 데이터를 받아오게 만들어야 한다. 여기서는 <input> 태그의 형식Type을 'text'로 넣었기 때문에 비밀번호를 화면에서 볼수 있지만, 만약 type 속성을 'password'로 변경하면 화면에서 읽는 것이 불가능하다.

③ '통계 보기' 버튼이 클릭되면 설문조사 데이터를 서버로부터 받아서 화면에 챠트로 통계를 보여준다.

④ '파일 저장' 버튼이 클릭되면 설문조사의 모든 내용이 엑셀Excel 파일로 만들어져서 다운로드된다.

⑤ 설문조사에 참여한 고객의 숫자는 페이지가 열리면 자동으로 받아서 화면에 보여지게 된다. 설문조사의 통계는 비밀번호를 가지고 서버로부터 요청하지만, 참여한 고객의 수는 서버에서 비밀번호를 확인하지 않고 데이터를 보내기 때문에 관리자 페이지가 브라우저에서 열릴 때 자동으로 화면에 보여진다.

⑥ 챠트를 만드는 영역은 태그를 사용해서 사각형 영역에 만들어지는데, 사각형의 크기는 'admin.css' 파일에 가로 300px, 세로 200px로 설정되었다. 'chart' 클래스 이름은 디자인을 위해서 사용되며, id 이름으로 각각의 챠트 영역으로 접근하게 된다. 챠트를 그리는 것은 이와 같이 로 만들어진 사각형의 태그 안에서 그림을 그리는 것이라고 이해하면 된다.

⑦ 챠트 오른쪽에 질문의 내용을 볼 수 있게 하기 위해서 설문조사의 내용을 그대로 사용하였다.

⑧ 이전의 태그들이 'float:left'로 정렬되었기 때문에, 다음 질문을 아래쪽에 배치하기 위해서 'clear:left'라는 속성을 가진 크기가 없는 태그를 사용했다. 이와 같이 하나의 속성을 가진 빈 태그를 사용해서 디자인 속성을 해제하는 방법은 아주 중요하게 사용되곤 한다.

[코드 3-6] 설문조사의 내용들을 통계로 보여주는 JavaScript 코드 (/admin/js/admin.js)

```
var dataPerson = new Array();                                    //---①

var aChart = new Array();                                        //---②

var arrNumQuestion = [ 6 , 3, 6, 2, 6, 7, 4, 5, 5 ];            //---③
```

```
window.onload = function() {                                    //---④

        getCountData();
};

function getCountData() {                                       //---⑤

        data = doAjax('ajax_get_count.php');                    //---⑥

        $('#count').html(data);

        setTimeout(getCountData,5000);                          //---⑦
}

function downloadFile() {                                       //---⑧

        var passwd = $("#passwd").val();

        window.location = './download_file.php?password='+passwd;  //---⑨
}

function drawChart() {                                          //---⑩

        var i, j, temp;

        var numQuestion = 9;                                    //---⑪
        var numAnswer = 10;                                     //---⑫

        var answer = new Array();                               //---⑬

        for (i=0; i<numQuestion; i++) {                         //---⑭
            answer[i] = new Array();
            for (j=0; j<numAnswer; j++) {
                        answer[i][j] = 0;
            }
        }

        for (i=0; i<dataPerson.length; i++) {                   //---⑮
            answer[0][dataPerson[i][2]-1] += 1;
            answer[1][dataPerson[i][3]-1] += 1;

            temp = dataPerson[i][4].split('&');
            for(j=0; j<temp.length; j++) {
                        answer[2][temp[j]-1] += 1;
```

```
            }

            answer[3][dataPerson[i][5]-1] += 1;
            answer[4][dataPerson[i][6]-1] += 1;

            temp = dataPerson[i][7].split('&');
            for(j=0; j<temp.length; j++) {
                        answer[5][temp[j]-1] += 1;
            }
            answer[6][dataPerson[i][8]-1] += 1;
            answer[7][dataPerson[i][9]-1] += 1;
            answer[8][dataPerson[i][10]-1] += 1;
        }

        var dataChart;

        for (i=0; i<numQuestion; i++) {                                    //---⑯

            dataChart = '[';                                               //---⑰
            for (j=0; j<arrNumQuestion[i]; j++) {
                        if (j != 0) { dataChart += ','; }
                        dataChart += '{x:'+(j+1)+',y:'+answer[i][j]+'}';
            }
            dataChart += ']';

            aChart[i] = new CanvasJS.Chart("chart"+(i+1),                  //---⑱
                                    {
                                                animationEnabled: true,    //---⑲
                                                data: [ {
                                                            type: "column",//---⑳
                                                            dataPoints: eval(dataChart)  //---㉑
                                                        } ]
                                    });

            aChart[i].render();                                            //---㉒
        }
}

function getAllData(){                                                     //---㉓

        var passwd = $("#passwd").val();
        var param = { password : passwd };
        data = doAjax('ajax_get_alldata.php', param);                      //---㉔

        if (data == 0) {                                                   //---㉕
            alert("Error: 올바른 비밀번호를 입력하세요.");
```

```
        return false;
    }

    var people = data.split('&&&&');                                    //---㉖

    for (var i=0; i<people.length-1; i++) {                             //---㉗
        dataPerson[i] = people[i+1].split('##');
    }

    drawChart();                                                        //---㉘
}

function doAjax(str_url, input_data) {
    var result;
    $.ajax({
        url: str_url,
                type: 'post',
                async: false,
                datatype: 'json',
                data: input_data,
                error: function() {
                        alert('Network Error : 서버와의 연결에 문제가 있습니다.');
                },
                success: function(obj) {
                        result = obj;
                }
    });
    return result;
}
```

① 설문조사의 모든 데이터를 가지고 있는 배열이 dataPerson이며, dataPerson 배열의 내용은 고객들의 데이터를 담고 있는 또다른 배열을 가지고 있다. 예를 들면, dataPerson 배열을 접근할 때 dataPerson[0]은 첫 번째 고객의 데이터를 가지고 있으며, dataPerson[i]라는 것은 'i+1'번째 고객의 데이터를 담고 있다. 또한, dataPerson[0][0]은 첫 번째 고객의 이름이고, dataPerson[0][1]은 첫 번째 고객의 전화번호이고, dataPerson[0][2]는 첫 번째 고객의 'q1' 질문에 대한 답을 가지고 있다. 이와 같이 배열 안에 또다른 배열을 사용함으로써 설문조사에 참여한 모든 고객의 정보를 dataPerson 배열에 담아서 프로그래밍이 진행된다. 서버에 저장된 데이터의 구조에 기반해서 '&&&&' 구분자로 각각의 고객을 나누고, '##' 구분자로 한 명의 고객 데이터를 나누어서 배열에 저장한다는 것을 이해하면 이와 같이 하나의 변수를 사용해서 모든 고객의 데이터를 저장할 수 있다는 것을 쉽게 이해할 수 있다.

② CanvasJS 라이브러리를 사용해서 챠트를 만들게 되면 생성되는 챠트도 하나의

객체이며 aChart라는 배열에 생성된 챠트 객체를 넣어주면 된다. 여기서는 생성된 챠트 객체에 다시 접근하지 않기 때문에 aChart라는 배열이 필요하지 않을 수 있으나, 많은 고급 웹 프로그래밍에서는 만들어진 챠트를 동적으로 만들기 위해서 챠트 객체에 데이터를 변경하는 작업을 진행한다. 화면에서 자동으로 움직이는 챠트를 본다면 챠트 객체에 접근해서 데이터를 변경하고 다시 그려주었다고 이해하면 된다. (다시 그려주는 작업을 프로그래밍 용어로는 렌더링Rendering이라고 부른다.)

③ 설문조사에서 선택하는 질문은 총 9개이며 각각의 질문에서 선택할 수 있는 갯수가 다르기 때문에 arrNumQuestion 배열을 사용해서 9개의 질문에서 선택해야하는 갯수를 명시했다. 즉, "arrNumQuestion[0] = 6"이라는 것은 첫 번째 질문에서 선택할 수 있는 항목들이 6개라는 것이며, "arrNumQuestion[5] = 7"이라는 것은 6번째 질문에서 선택할 수 있는 항목들이 7개로 만들어졌음을 코드에 기록한 것이다.

④ <body> 태그에서 onload 이벤트를 사용해서 모든 HTML의 내용을 받은 후에 자동으로 실행되는 함수를 지정할 수 있다. 이와 같은 동작을 하는 JavaScript 언어의 예약어로는 'window.onload' 라는 것이 있는데, 'window.onload'를 함수로 만들어서 브라우저 전체의 내용을 받은 이후에 자동으로 실행되는 JavaScript 코드를 만들 수 있다. 즉, 관리자 페이지가 브라우저에서 열리면 'window.onload' 이벤트가 발생해서 getCountData() 함수를 자동으로 실행하게 된다.

⑤ 설문조사에 참여한 고객의 숫자를 서버로부터 받아와서 화면에 '참여 고객수'를 보여준다.

⑥ '참여 고객수'를 서버로부터 가져올 때 서버의 'ajax_get_count.php' 파일을 실행하며, 서버로 보내는 데이터는 없다. 서버로부터 받은 정보는 'count' id 값을 가진 태그에 넣어준다.

⑦ 관리자가 화면을 보고있는 중간에도 설문조사에 참여한 고객이 있다면 실시간으로 '참여 고객수'가 변경되기 때문에 5초마다 반복해서 '참여 고객수'의 숫자를 서버로부터 가져오기 위해서 setTimeout() 함수를 사용했다.

⑧ '파일 저장' 버튼이 선택되면 실행되는 함수가 downloadFile() 함수이며, 고객의 모든 데이터를 엑셀 파일로 만들어서 다운받는다.

⑨ 서버의 'download_file.php' 파일을 실행시키기 위해서 비밀번호를 'Get 방식'으로 서버로 보내고 엑셀 파일을 다운받는다. 파일을 다운받는 방법은 'window.location' 예약어를 사용하는데, 브라우저에서는 이와 같은 방법으로 파일을 다운받게 된다.

⑩ 챠트를 그리는 함수이며 dataPerson 배열에 저장된 모든 데이터를 가지고 챠트

에 필요한 통계 데이터를 만든 후에 챠트를 그려준다.

⑪ 만들기 위한 챠트는 9개이며, 9개의 질문의 숫자를 numQuestion에 저장했다. 이와 같이 사용하는 이유는 이후에 질문의 갯수가 변경되어 만들기 위한 챠트가 많아져도 numQuestion의 갯수만 변경하면 된다.

⑫ 각각의 질문에 대한 선택항목의 최대 갯수를 10으로 정하였는데, 질문에 대한 답을 할 수 있는 항목의 갯수가 다양하기 때문에 최대값을 10으로 정해서 프로그래밍을 단순화 시키기 위해서 사용하였다.

⑬ 챠트를 만들기 위해서 가장 중요한 것이 answer 배열을 이해해야 하는데, dataPerson 배열과 비슷한 구조를 갖는다. 예를 들면, answer[0]은 첫 번째 질문에 대한 항목들의 값들을 가지고, answer[i]는 'i+1'번째 질문에 대한 답의 결과를 저장한다. 또한, answer[0][0]은 첫 번째 질문의 첫 번째 항목을 선택한 고객의 숫자를 저장하고, answer[0][2]은 첫 번째 질문에서 세 번째 항목을 선택한 고객의 숫자를 저장한다. 챠트에 보여지는 데이터는 각각의 질문에 대한 해당 선택 항목들을 몇 번 선택되었는지를 보여주는 것이기 때문에, 통계 데이터를 만들어서 answer 배열에 저장한다.

⑭ answer 배열의 모든 내용들을 0으로 초기화한다. 'answer' 배열은 1차원 배열이지만, 'answer[i] = new Array();'를 사용해서 2차원 배열로 만들어진다. 재미있는 것은 배열 안의 값을 배열로 만드는 것이 가능하다는 것은 프로그래밍의 세계에서는 수학의 세계와 마찬가지로 무한의 차원을 만드는 것이 가능하다. 여담이라면, 물질의 세계에서는 이제서야 5차원을 논하고 있지만, 수학과 프로그래밍에서는 무한차원을 논할 수가 있다.

⑮ 고객의 데이터를 for loop을 사용해서 하나의 고객 데이터마다 차례대로 접근해서 통계 데이터를 만든다. 여기서 재미있는 것이 "answer[0][dataPerson[i][2]-1] += 1;"과 같이 사용해서 고객이 선택한 항목의 숫자를 1씩 증가하는 것인데, 프로그래밍이 컴퓨터 게임보다 재미있다는 것이 이런 수식을 만드는 재미가 있기 때문이다. "dataPerson[i][2]" 라는 것은 'i+1'번째 고객이 첫 번째 질문인 'q1'에 대한 선택값을 저장하고 있는데, "dataPerson[i][2] – 1"의 값은 첫 번째 질문을 의미하는 answer[0]의 선택된 항목의 배열의 위치를 의미한다. 예를 들면, 첫 번째 고객이 첫 번째 질문에서 2번재 항목을 선택했다면 dataPerson[0][2]는 2의 값을 갖으며, 통계를 위한 answer 배열에서 첫 번째 질문의 2번째 항목을 의미하는 answer[0][1]의 값에 1을 더한다. 다소 복잡해보이는 수식이지만 여러 줄로 만들 수 있는 코드를 한 줄로 간

략히 만들었으며, 이러한 방법이 코딩의 묘미 중의 하나일 수 있다. 이러한 방법으로 통계를 위해서 만든 answer 배열의 모든 값들을 계산할 수 있으며, 다중 선택이 가능한 3번째와 6번째 항목에서는 '&' 구분자를 사용해서 중복된 선택을 모두 처리할 수 있다. 문자열을 처리하기 위하여 많이 사용되는 것이 split() 함수인데, split() 함수에 구분자를 넣어서 실행하면 배열로서 구분된 값들을 받게된다.

⑯ answer 배열에 통계데이터가 만들어졌고, 여기서는 9개의 챠트를 만들기 위해서 for loop을 사용해서 챠트를 하나씩 만든다.

⑰ 챠트에 넣는 데이터는 배열이며 dataChart 변수는 배열의 형태를 갖는 문자열이다. 배열은 다수의 값을 저장하고 있지만, 문자열로 만들어진 dataChart 변수는 단지 문자들의 조합일 뿐이다. 그래서, 이후에 챠트에 넣는 dataChart 변수의 값은 수식으로 변환작업을 해야 한다. 배열 형식의 문자열을 만들기 위해서 해당 질문에 대항 항목의 갯수를 저장하고 있는 arrNumQuestion 배열의 값만큼 반복해서 배열 안의 값을 만들어준다. 문자열을 만들 때 두 번째 항목부터는 앞에 ','(콤마)를 사용해서 구분을 지어준다. 배열 안의 값은 JSON 데이터 형식으로 만들어주어야 하며, JavaScript 언어에서는 JSON 데이터를 참 많이 사용하고 있다. 챠트에서 x-좌표는 선택되는 항목의 갯수를 나타내기 때문에 1부터 순차적으로 증가하고, y-좌표는 선택 항목의 통계 값을 갖는다. 챠트를 그리기 위해서 배열과 JSON 데이터의 구조를 확실히 이해하고 있어야 하며, 이와 같이 배열과 JSON 데이터를 혼용해서 사용하는 방법을 많은 라이브러리에서 사용하고 있다.

⑱ CanvasJS 챠트를 만들기 위해서는 CanvasJS.Chart() 생성자를 사용해야 하며, new를 사용해서 생성하기 때문에 CanvasJS.Chart()는 클래스 개념으로 만들어 졌음을 알수 있다. 또한 new를 사용해서 객체를 생성한다는 것은 메모리에 공간을 차지한다는 것으로 이해하면 된다. Chart() 생성자에는 두 개의 매개변수가 필요한데, 첫 번째는 화면에서 그려주기 위한 사각형의 위치 객체의 id 값을 넣어주고, 두 번째는 챠트의 속성과 데이터 값들을 JSON 데이터 형식으로 넣어준다. 예를 들면, 첫 번째 챠트를 만들기 위해서는 i가 0의 값을 갖기 때문에, 첫 번째 매개변수는 'chart1'이라는 값이 넣어지는 것이고, HTML 코드에서 첫 번째 챠트를 위해서 만들어진 태그이 id 값이 'chart1'이었기 때문에 여기에 현재의 챠트가 그려진다. 생성된 챠트 객체는 전역변수로 만들어진 aChart 배열에 넣어지는데, 만약 이후에 이 챠트의 객체에 접근할 필요가 있으면 aChart 배열을 통해서 해당 챠트를 제어하면 된다.

⑲ 챠트가 그려질 때 움직이는 효과를 주기 위해서 animationEnabled 속성을 true로 설정한다. 이렇게 하면 챠트가 그려지는 순간에 부드럽게 움직이는 효과를 주게 되는데, CanvasJS 라이브러리에서 제공하는 디자인 속성이다.

⑳ 챠트의 형식Type을 정하기 위해서 type 속성을 정해야 하는데, 여기서는 막대그래프를 만들기 위해서 'column' 형태의 챠트를 만든다. 이와 같이, CanvasJS 라이브러리에서 제공하는 다양한 함수를 사용하기 위해서는 type 속성의 값을 변경하면 된다.

㉑ dataChart는 챠트에 넣어지는 데이터 값인데, 문자열로 만들어졌기 때문에 수식으로 변경하기 위해서 eval() 함수를 사용해서 문자열을 배열로 변경한다. 이와같이 문자열을 수식으로 변경하는 함수가 eval() 함수인데 외부로부터 받은 데이터를 가지고 eval() 함수를 사용하면 보안에 아주 취약하기 때문에 위험한 함수로 받아들여지곤 한다.

㉒ aChart[i]는 챠트 객체를 가지고 있고, 클래스로 만들어진 챠트 객체의 render() 함수를 사용해서 화면에 챠트를 그린다. 이와 같이, 대부분의 챠트 라이브러리는 챠트를 그려주는 함수를 가지고 있으며, 데이터가 변경된 이후에는 render() 함수와 같이 다시 그려주는 작업을 진행해야 한다.

㉓ '통계 보기' 버튼이 클릭되었을 때 실행되는 함수로서, 설문조사의 모든 데이터를 가져와서 dataPerson 배열에 넣고 챠트를 그려주는 drawChart() 함수를 실행한다.

㉔ 비밀번호를 서버에 전달하고 'ajax_get_alldata.php' 함수를 실행해서 설문조사의 모든 데이터를 받아온다.

㉕ 서버로 받은 데이터 값이 0이면 비밀번호가 맞지 않았다는 것을 의미하는 경고창을 출력한다.

㉖ 서버로부터 받은 데이터를 고객 한 명의 데이터들로 구분해서 people이라는 배열에 저장한다. JavaScript 언어의 함수 중에 문자열을 나누는 split() 함수는 특정 문자열을 기준으로 전체 문자를 분리해서 배열로 만든다. 여기서는 고객의 구분자인 '&&&&'를 사용해서 개별 고객 데이터로 구분하였다. 고객 데이터의 가장 앞쪽에는 항상 '&&&&' 구분자가 있기 대문에 배열로 만들어진 첫 번째 people[0]에는 빈문자열이 존재한다.

㉗ 한 명의 고객 데이터를 dataPerson 배열에 '##' 구분자를 사용해서 배열로 만들어서 넣는다.

㉘ 모든 데이터를 서버로부터 받아서 dataPerson 배열에 넣은 이후에 이 데이터들

을 가지고 챠트를 그려주기 위해서 drawChart() 함수를 실행한다.

관리자 페이지에서 설문조사에 대한 통계를 보여주기 위해서 브라우저에서는 JavaScript 코드가 많은 연산을 수행한다. 서버에서 데이터를 JSON 데이터로 만들어서 전송해주어도 괜찮지만, 서버에 부하를 최소화하기 위해서 서버에서는 파일의 데이터를 읽어서 브라우저로 전송하는 역할만 하도록 구현하였다.
[코드 3-7]과 [코드 3-8]은 서버에서 동작하는 PHP 코드들을 보여주는데, 브라우저로부터 요청을 받고 단지 파일에서 데이터를 읽어서 브라우저로 전송만 하는 역할을 하도록 구현하였다.

[코드 3-7] 설문조사에 참여한 고객의 수를 보내주는 PHP 코드 (/admin/ajax_get_count.php)

```php
<?php

$filename = "../2016/data/count.data";                                    //---①
$handle = fopen($filename, "r");
$data = fread($handle, filesize($filename));
fclose($handle);

print $data;
exit();

?>
```

① '참여 고객수'가 저장된 'count.data' 파일에서 숫자를 읽고 브라우저로 전송한다.

[코드 3-8] 설문조사의 모든 데이터를 보내주는 PHP 코드 (/admin/ajax_get_alldata.php)

```php
<?php

$password;

if(isset($_POST['password'])) { $password = $_POST['password']; }

if ($password != 'jisiksoft') {                                           //---①
        print 0;
        exit();
}

$filename = "../2016/data/alldata.data";                                  //---②
$handle = fopen($filename, "r");
$data = fread($handle, filesize($filename));
fclose($handle);
```

```
print $data;
exit();

?>
```

① 'Post 방식'으로 받은 비밀번호를 확인하고, 일치하지 않으면 'Error'를 의미하는 0을 브라우저로 전송하고 실행을 마친다.

② 비밀번호가 일치하였다면, 'alldata.data' 파일에서 설문조사의 모든 데이터를 읽어서 브라우저로 전송한다.

설문조사의 데이터는 구분자('&&&&', '##', '&')를 가지고 문자열을 조작하는 방법으로 구현하였는데, 독자들은 JSON 데이터로 만드는 것도 좋은 연습이라고 생각한다. 프로그래밍을 하다보면 다른 사람의 코드를 단순히 복사해서 사용하는 경우가 많은데, 많은 시간이 소요되더라도 자신이 사용하는 코드를 정확히 이해하는 과정을 거치면 나중에 자신만의 멋진 프로그램을 만들 수 있다. 이번 장에서는 챠트를 만드는 방법을 간략히 소개했으나, 대부분의 챠트 라이브러리들은 많은 속성들을 만들어서 라이브러리를 사용하는 사용자가 원하는 대로 이미지를 변경할 수 있는 방법을 제공한다. 예를 들면, 디자이너가 챠트의 막대그래프 색의 변경을 원한다면 프로그래머는 라이브러리의 속성들을 사용해서 특정 색으로 그래프를 변경하는 것도 가능하다.

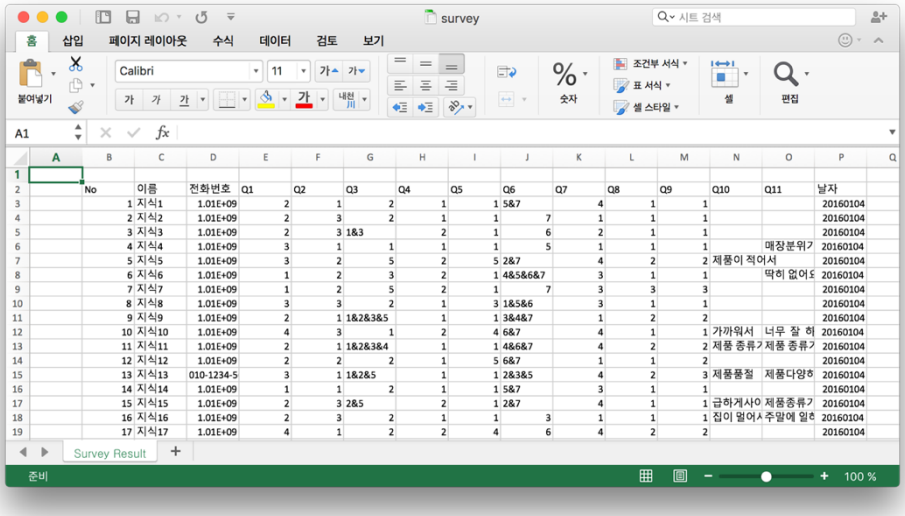
3.4 결과를 Excel 로 가져오기

설문조사의 데이터를 우리가 많이 사용하는 엑셀 문서로 만들어서 읽기 쉽게 만드는 것이 중요하다. 프로그래밍을 한다는 것은 사용자에게 편리한 프로그램을 만드는 것이 중요하기 때문에, 서버에 저장된 데이터를 설문조사의 고객인 매장–대표가 쉽게 볼수 있는 환경을 만들어주는 것이 중요한다. 브라우저에서 JavaScript 언어를 사용해서 엑셀 파일로 만들어주는 것도 가능하고, 서버에서 엑셀 파일로 만들어서 다운로드 할 수 있게 만드는 것도 가능하다. 많은 사용자가 접속하는 서버라면 브라우저에서 엑셀 파일을 만드는 것이 좋으나, 설문조사에서는 다운받는 사람들은 소수에 불과하기 때문에 이 책에서는 서버에서 PHP 언어로 엑셀파일을 만들어서 다운받을

수 있게 만들었다. 특정 문서로 변환을 한다는 것은 정해진 규격에 맞추어서 데이터를 만들어주는 변환작업을 해주는 것인데, 이것을 직접 만들어서 사용하기 위해서는 엑셀 파일의 헤더Header 부분과 내용 부분의 저장되는 형식을 분석해야 한다. 그러나, 다행히도 이전 장의 챠트 라이브러리와 마찬가지로 PHP에서도 엑셀 파일을 만들수 있는 'PHPExcel' 라이브러리가 존재한다. 일반적으로 라이브러리들을 접하는 사용자들은 이것도 공부해야 하나 하고 생각하지만, 직접 만들기 위해서 소요되는 시간을 생각하면 1/100로 시간이 단축된다고 이해하는 것이 좋다. 또한, 인터넷에는 라이브러리를 사용하는 예제들이 많이 있기 때문에 아주 짧은 시간에 라이브러리를 활용하는 법을 알 수 있다. [그림 3-8]은 관리자 화면에서 '파일 저장' 버튼을 클릭해서 다운받은 엑셀 파일을 확인한 그림이며 모든 데이터가 각각의 셀Cell에 저장되어 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_3/3.4/survey/admin/ (비밀번호: 'jisiksoft')

그림 3-8 엑셀 파일로 설문조사 내용을 다운받아서 확인한 결과



라이브러리를 만드는 회사에서는 사용자로부터 비용을 받아야 운영이 되는데, 무료는 홍보를 위해서고 상업용으로 사용하는 사용자로부터 일정 비용을 받는 구조가 보편화되었다. 그러나, 많은 라이브러리들이 무료로 사용할 수 있게도 만들어졌는데, 1990년대부터 본격화된 오픈소프 프로젝트 중의 하나인 GNU 프로젝트로 인해 PHPExcel 라이브러리도 GNU의 정책을 따른다.

[코드 3-9]는 'PHPExcel.php' 파일에서 가장 먼저 설명하는 라이센스 정책을 보여주는데, 누구나 무료로 사용하고 배포할 수 있다고 나와 있다. 서버에서 PHP 언어를 사용해서 엑셀 파일을 쉽게 만들 수 있게 'PHPExcel' 라이브러리를 만들어 준 GNU

프로젝트 담당자에게 감사드린다.

[코드 3-9] PHPExcel 라이브러리의 오픈소스 라이센스 설명 (/lib/PHPExcel/Classes/PHPExcel.php)

```
<?php
/**
 * PHPExcel
 *
 * Copyright (c) 2006 - 2013 PHPExcel
 *
 * This library is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * This library is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.    See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with this library; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA    02110-1301    USA
 *
 * @category     PHPExcel
 * @package      PHPExcel
 * @copyright    Copyright (c) 2006 - 2013 PHPExcel (http://www.codeplex.com/PHPExcel)
 * @license      http://www.gnu.org/licenses/old-licenses/lgpl-2.1.txt       LGPL
 * @version      1.7.9, 2013-06-02
 */

##### 코드 생략 #####
```

[코드 3-10]은 PHPExcel 라이브러리의 함수들을 사용해서 엑셀 파일을 만드는 PHP 코드이며, 필자도 인터넷 검색으로 사용법을 알게 되었다. 모든 것을 완벽히 설명하는 자료는 사실 드물기 때문에 독자분들도 자신이 궁금해하는 부분이 있으면 인터넷 검색으로 좀더 깊이 있는 내용을 찾기를 바란다. [코드 3-10]에서는 PHPExcel 라이브러리를 사용하기 위해서 'require_once' 예약어로 'PHPExcel.php' 파일을 가져온 후 PHPExcel() 생성자를 사용해서 클래스를 생성한 후에 클래스 안의 함수Method들을 사용하는 것을 보여준다.

[코드 3-10] PHPExcel 을 이용해서 데이터를 엑셀파일로 만드는 PHP 코드 (/admin/download_file.php)

```
<?php
```

```
$password;
if(isset($_GET['password'])) { $password = $_GET['password']; }

if ($password != 'jisiksoft') {                                          //---①
        exit();
}

$filename = "../2016/data/alldata.data";                                 //---②
$handle = fopen($filename, "r");
$strData = fread($handle, filesize($filename));
fclose($handle);

$arrPeople = split("&&&&", $strData);                                    //---③

require_once './lib/PHPExcel/Classes/PHPExcel.php';                      //---④

$objPHPExcel = new PHPExcel();                                           //---⑤

$objPHPExcel->getProperties()->setCreator("jisiksoft.com")               //---⑥
                             ->setLastModifiedBy("jisiksoft.com")
                             ->setTitle("Survey Result")
                             ->setSubject("Survey Result 2016")
                             ->setDescription("Survey Result 2016")
                             ->setKeywords("office 2007 openxml php")
                             ->setCategory("Survey Result 2016");

$objPHPExcel->setActiveSheetIndex(0)                                     //---⑦
                        ->setCellValue('B2', 'No')
                        ->setCellValue('C2', '이름')
                        ->setCellValue('D2', '전화번호')
                        ->setCellValue('E2', 'Q1')
                        ->setCellValue('F2', 'Q2')
                        ->setCellValue('G2', 'Q3')
                        ->setCellValue('H2', 'Q4')
                        ->setCellValue('I2', 'Q5')
                        ->setCellValue('J2', 'Q6')
                        ->setCellValue('K2', 'Q7')
                        ->setCellValue('L2', 'Q8')
                        ->setCellValue('M2', 'Q9')
                        ->setCellValue('N2', 'Q10')
                        ->setCellValue('O2', 'Q11')
                        ->setCellValue('P2', '날자');

for ($i=1; $i<count($arrPeople); $i++) {                                 //---⑧
```

```
        $data = split("##", $arrPeople[$i]);                                    //---⑨
        $objPHPExcel->setActiveSheetIndex(0)
                    ->setCellValue('B'.($i+2), $i)
                    ->setCellValue('C'.($i+2), $data[0])
                    ->setCellValue('D'.($i+2), $data[1])
                    ->setCellValue('E'.($i+2), $data[2])
                    ->setCellValue('F'.($i+2), $data[3])
                    ->setCellValue('G'.($i+2), $data[4])
                    ->setCellValue('H'.($i+2), $data[5])
                    ->setCellValue('I'.($i+2), $data[6])
                    ->setCellValue('J'.($i+2), $data[7])
                    ->setCellValue('K'.($i+2), $data[8])
                    ->setCellValue('L'.($i+2), $data[9])
                    ->setCellValue('M'.($i+2), $data[10])
                    ->setCellValue('N'.($i+2), $data[11])
                    ->setCellValue('O'.($i+2), $data[12])
                    ->setCellValue('P'.($i+2), $data[13]);
}

$objPHPExcel->getActiveSheet()->setTitle('Survey Result');                      //---⑩

header('Content-Type: application/vnd.ms-excel;charset=utf-8');                 //---⑪
header('Content-Type: application/x-msexcel;charset=utf-8');
header('Content-Disposition: attachment;filename="survey.xls"');
header('Cache-Control: max-age=0');

$objWriter = PHPExcel_IOFactory::createWriter($objPHPExcel, 'Excel5');          //---⑫
$objWriter->save('php://output');

exit();

?>
```

① 'Get 방식'으로 받은 비밀번호가 올바르지 않으면 프로세스를 진행하지 않는다.

② 설문조사의 모든 데이터를 'alldata.data' 파일로부터 읽어온다.

③ '&&&&' 구분자를 기준으로 split() 함수를 사용해서 고객별 데이터로 분리해서 $arrPeople 배열에 저장한다.

④ PHPExcel 라이브러리를 사용하기 위해서 'PHPExcel.php' 파일을 연결한다.

⑤ $objPHPExcel 변수는 PHPExcel() 클래스 생성자로 만들어진 객체Object를 가지게 되는데, PHPExcel 라이브러리는 코드를 직접 보지 않아도 PHP 언어를 사용해서 클래스 개념으로 만들어졌음을 알수 있다.

⑥ 생성된 PHPExcel 클래스의 속성을 'getPrroperties()' 함수를 사용해서 가져온 후에 세부정인 항목들을 설정한다.

⑦ PHPExcel에서는 엑셀의 시트Sheet를 가져와서 특정 위치의 셀Cell에 값을 넣는 작업으로 진행하는데, setActiveSheetIndex() 함수를 사용해서 엑셀의 첫 번째 시트Sheet를 정의한다. 이후에는 해당 시트Sheet의 특정 위치의 셀에 setCellValue() 함수를 사용해서 값을 넣는다. 엑셀의 표는 가로의 구분은 알파벳으로 이루어지고, 세로는 숫자로 구분된다. 즉, "setCellValue('D2', '전화번호')"를 실행하면, D-열의 2-행에 위치한 셀Cell에 '전화번호'라는 문자를 넣는다.

⑧ 고객의 데이터들을 for loop을 사용해서 한 고객의 데이터를 차례대로 엑셀의 셀에 넣는다.

⑨ $data 변수는 배열로 사용되는데 한 고객의 개별 내용들을 배열에 가지고 있으며, 각각의 셀에 setCellValue() 함수를 사용해서 값들을 넣어준다.

⑩ 데이터가 들어간 셀의 이름을 설정하기 위해서 setTitle() 함수를 사용했으며, 다운받은 엑셀 파일의 아랫부분에 있는 시트Sheet의 이름이 'Survey Result'로 보이게 된다.

⑪ HTTP 프로토콜로 브라우저로 전송을 하기 위해서 header()를 만들어주는 것이며, 다운받을 수 있게 첨부된 파일의 이름이 'survey.xls'라는 엑셀파일이라고 브라우저에게 알려주게 된다.

⑫ PHPExcel로 만들어진 객체는 메모리에 저장되어 있기 때문에 createWriter() 함수를 사용해서 메모리에 있는 객체의 내용을 엑셀 파일의 형식으로 변환해주고, save() 함수를 사용해서 브라우저에 전송한다.

지금까지 설문조사를 할수 있는 페이지를 만들고, 설문조사의 내용을 확인할 수 있는 관리자 페이지를 만들었다. 관리자 페이지를 만들면서 이해해야 할 것은 웹 프로그래밍을 할 때는 다양한 라이브러리들을 사용하는 것이 좋으며, 비용이 다소 발생하더라도 시간을 줄일 수 있다는 측면에서 라이브러리를 사용하는 것이 아주 좋다. 웹 프로그래밍을 한다는 것은 실제 많은 시간을 할애해야 하는 경우가 많기 때문인데, 화면에서 디자인하기 위해서 소요되는 시간도 많고 또한 브라우저와 서버 쪽에서 동시에 프로그래밍을 진행해야 하기 때문이기도 하다. 하지만, 기본적인 개념을 이해하고 조금씩 프로그램을 만들어 간다고 생각하고 긴 시간의 프로그래밍 작업을 진행하기를 바란다. 간혹, 실무에서 일하는 사람들 중에서 JavaScript와 JSP의 차이를 이해하지 못하는 사람을 볼 때가 있다. JavaScript는 브라우저에서 동작하는 언어이고, JSP는 Java 언어를 사용하는 PHP와 같은 서버쪽 프로그래밍이다. 그런데, 사이

트를 만들다 보면 브라우저에서 실행하는 JavaScript 코드를 JSP 안에서 만드는 작업이 많이 필요한데, JavaScript는 브라우저에서 동작하는 프로그래밍 언어라는 것을 이해하면 JSP에서 구조적으로 분리해서 프로그래밍을 하기가 수월해질 수 있다. 이 책에서는 PHP 언어를 간단히 보여주지만, JSP도 PHP에서 동작하는 원리와 같은 개념으로 서버에서 동작하는 Java 기반의 플랫폼을 사용한다고 이해하면 된다.

chapter 4

웹 블로그 만들기

많은 사람들이 포털 사이트의 블로그를 사용하는데, Daum과 Naver의 블로그가 대표적이다. 포털 사이트들은 많은 사람들이 사용하는 대국민 서비스를 하고 있는 것이며, 다른 사람과의 정보를 공유하면서 인터넷 세상에서 정보를 생산하는 역할을 하고 있다. 포털에서 제공하는 블로그를 사용하면 웹 프로그래밍을 굳이 배울 필요가 없으며, 자신이 잘 알고 있는 내용을 가지고 인터넷 세상에서 자신만의 공간을 쉽게 가질 수 있다. 그런데, 웹 프로그래밍을 하는 필자의 입장에서는 포털에서 만드는 나의 정보가 내 것만이 아닌 느낌을 받는다. 예전에 활성화 되었던 '싸이월드'나 '프리첼' 같은 사이트에 등록된 개인의 글들도 서비스를 제공하는 회사가 기존 서비스의 운영을 중단하면 개인의 정보는 인터넷 상에서 없어지게 된다. 포털에 등록한 나의 정보는 타인이 접근하기 아주 좋은 환경을 만들어 주지만, 반대로 개인 정보는 포털 사이트의 중요한 홍보 수단으로 사용될 수도 있다. 포털을 사용하다보면 검색의 우선 순위는 포털 회사의 데이터베이스에 저장된 내용이 우선 순위에 있으며, 좀더 객관적으로 인터넷 세상의 정보를 보여주는 구글보다는 다소 편향된 검색결과가 나오기도 한다. 특히나 컴퓨터 프로그래밍을 하는 필자와 같은 경우는 한국의 포털이 구글의 영어 사이트에서 정보를 찾아 단순히 한글로 번역하여 소개한 내용들이 많은 것을 보게 되는데 그래서 프로그래밍을 하는 사람들에게 영어로 구글에서 검색하는 것을 권하기도 한다. 이번 Chapter에서 만들게 되는 '웹 블로그'는 인터넷 세상에서 자신의 개인 공간을 갖는 방법을 구현하며, 구글에서는 검색이 되지만 한국의 대표적인 포털 사이트들에는 검색이 안 될 가능성도 있다. 하지만, 인터넷 세상에서 자신이 만든 도메인(인터넷 주소)으로 자신만의 세계를 만드는 것은 상당히 재밌음을 알게 될 것이다. 여기서 만드는 블로그는 현재 '미들스토리'(http://middlestory.com) 라는 이름으로 운영 중인 블로그 형태의 홈페이지를 만드는 방법을 설명한다. 블로그는 한 번만 구축하고 나면 이후부터의 관리는 문서를 서버에 등록하기만 하면 되며, 생각보다 아주 간단하게 운영할 수 있기 때문에 현재

'미들스토리' 운영자는 프로그래밍에 대해서 전혀 모르지만 스스로 '미들스토리'에 글을 쉽게 등록하고 있다. 인터넷 상에서 자신만의 도메인으로 블로그 형태의 사이트를 운영하려는 사람들은 생각보다 많은데, 자신의 학술내용을 홍보하거나 소설이나 만화와 같은 글들을 독자들에게 자주 공개해야 하는 사람들에게 아주 유용하다. 웹에서 블로그 형태의 사이트를 만들기 위해서는 가장 먼저 해야할 것은 자신의 도메인을 갖는 것인데, '.com' 도메인을 좋아하는 필자 같은 경우에는 GoDaddy(http://kr.godaddy.com) 라는 외국 사이트를 사용하고 있다. '.co.kr' 도메인을 갖기를 원한다면 한국에서 운영하는 다수의 사이트들 중에서 선택해서 이용하면 된다. (ex: http://whois.co.kr, http://cafe24.com) 자신만의 도메인을 만들어 서버에 블로그 사이트를 구축한 이후에는 도메인을 등록한 회사에서 도메인 설정의 변경을 통해서 서버로 연결하면 된다. 현재 필자는 한 대의 클라우드 서버를 가지고 다수의 도메인에 대한 서비스를 진행하고 있는데, 대형 호스팅 업체들도 한 대의 서버에 다수의 홈페이지를 만들어서 서비스를 하고 있는 것으로 이해하면 된다. 서버에서 다수의 도메인에 대한 서비스를 위해서는 구글에서 "리눅스 서버 멀티 도메인"이라고 검색하면 리눅스에서의 설정 방법들을 찾을 수 있다.

[그림 4-1]은 이번 Chapter에서 만드는 블로그 형태의 사이트를 보여주는데, 일반적인 사이트와 달리 '로그인'하는 부분이 없으며 방문객은 아래의 왼쪽에 오늘 방문한 숫자(Today)와 그동안 방문한 전체 방문객 숫자(Total)를 보여준다. 또한 게시판이 없고 왼쪽에서 연도와 날짜를 클릭해서 블로그 내용들을 볼 수 있도록 구성했다. 물론, 이 책을 이해한 독자는 자신만의 새로운 디자인으로 만드는 방법을 이해하게 될 것이다. 일반 홈페이지와 다르게 웹 블로그를 설명하는 이유는 대부분의 홈페이지들은 데이터베이스Database를 사용해서 좀더 복잡한 구조로 만들지만, 여기서 다루는 웹 블로그는 데이터베이스를 사용하지 않고 모든 것을 서버에서 파일로 내용을 만들어서 관리하게 된다. 즉, 이번 Chapter에서는 웹 프로그래밍에서 데이터베이스를 사용하지 않고도 쉽게 관리할 수 있는 사이트를 만드는 방법을 설명한다.

확인 사이트: http://middlestory.com

그림 4-1 'http://middlestory' 개인 블로그 사이트

4.1 화면 분할

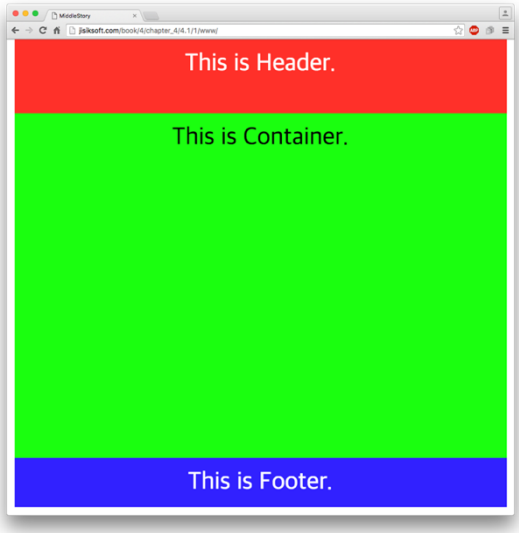
브라우저에서 사이트를 만들기 위해서는 화면 분할을 먼저 해야하는데, 전체 화면을 큰 단위로 나누어서 각각의 부분에 필요한 내용을 넣으면 된다. 오래전에는 화려한 이미지를 만들어서 사이트에서 보여주는 방법이 주로 사용되었지만, 수정하기 위해서는 이미지 편집을 해야하는 작업이 상당히 번거로웠다. 왜냐하면, 글자 하나를 변경하기 위해서는 이미지 편집 프로그램을 사용해야 하기 때문인데, 현재는 텍스트 기반의 사이트를 만드는 것을 더 선호하고 있으며, 화면의 분할을 통해서 해당 부분의 내용만을 간단히 수정할 수 있고 디자인을 다양하게 만드는 방법도 많이 발전했다.

[그림 4-2]는 블로그 사이트를 만들기 위해서 전체 화면을 분할했는데, 바탕색을 다

르게 해서 분할된 영역을 보기 쉽게 하였다. '미들스토리' 사이트와 비교하면 전체 메뉴가 들어가는 부분이 'Header' 영역이고, 블로그의 내용은 'Container' 영역에 넣게 되며, 화면의 아래에 항상 같은 내용을 보여주는 항목이 'Footer' 영역에서 만들어진다.

확인 사이트: http://jisiksoft.com/book/4/chapter_4/4.1/1/www/

그림 4-2 웹 블로그를 만들기 위하여 화면을 분할하였다.



[코드 4-1]은 태그와 CSS 디자인으로 분할된 화면을 보여주는 코드이며, 블로그의 분할을 이해하기 위한 가장 간단한 화면이다.

[코드 4-1] 화면을 분할하기 위하여 만들어진 HTML 파일 (/1/www/index.html)

```
<!DOCTYPE html>

<head>
    <title>MiddleStory</title>
    <meta charset="utf-8"></meta>
     <meta name="keywords" content="미들스토리, middlestory, middle, story"></meta>
    <link rel="stylesheet" href="./css/middlestory.css"></link>
</head>
<body>
    <div id="header">This is Header.</div>                                    //---①
    <div id="container">This is Container.</div>
    <div id="footer">This is Footer.</div>
</body>
</html>
```

① 태그를 사용해서 세 개의 분할된 화면(header, container, footer)을

만들었다.

[코드 4-2] 분할된 화면의 디자인을 위한 CSS 파일 (/1/www/css/middlestory.css)

```
body {                                                    //---①
        width:1000px;
        margin:auto;
        font-size: 50px;
        text-align: center;
        line-height: 100px;
}
#header {
        position: relative;                               //---②
        min-height: 150px;                                //---③
        color: #ffffff;
        background-color: #ff3333;
}
#container {
        position: relative;
        min-height: 700px;
        color: #000000;
        background-color: #33ff33;
}
#footer {
        position: relative;
        min-height: 100px;
        color: #ffffff;
        background-color: #3333ff;
}
```

① 브라우저의 전체에 대한 CSS 디자인을 <body> 태그에서 정의한다.

② 세 개의 태그들은 위치Position 속성을 'relative'로 설정했으며, 1.2장의 CSS에서 'position' 속성에 대하여 자세히 설명하였다.

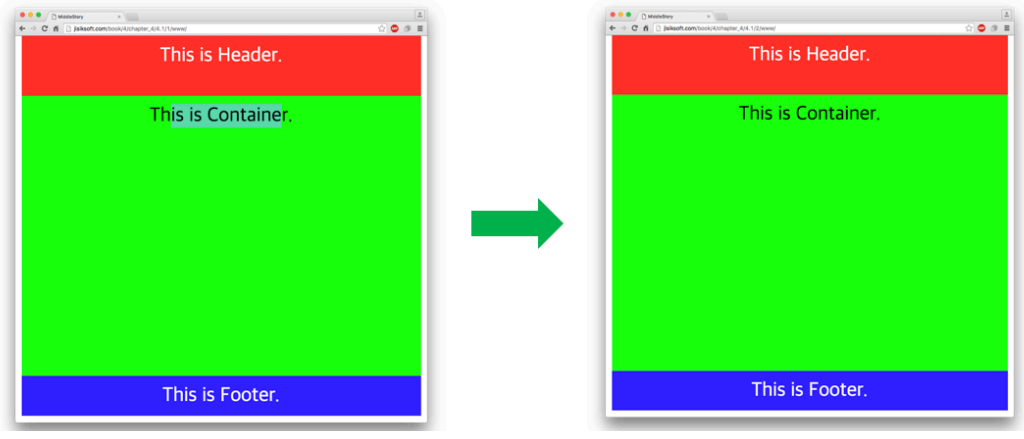
③ 'min-height'와 같이 높이(height)에 'min'이나 'max'를 앞에 붙여서 크기에 제한을 둘 수 있는데, 'min-height: 150px;'과 같이 설정하면 높이가 150px 이하로 작아지지 않으며 내용이 추가되면 높이가 길어질 수는 있다. 즉, 'min'을 사용해서 높이의 최소값이 정해졌다.

지금까지 만든 간단한 페이지에서 주의 깊게 볼 사항은 브라우저에서 마우스로 텍스트의 블럭Block 처리가 가능하다는 것인데, 이것은 브라우저에서 마우스의 클릭 이벤트를 받아들이기 때문이다. 지금까지 마우스의 클릭 이벤트는 'onmouseclick'으로 처리하였는데, 마우스를 클릭한다는 것은 실제 마우스 버튼을 누르고 띄었을 때의

두 개의 동작으로 구분할 수 있다. 즉, 브라우저에서 화면에서 글자를 블럭Block 처리한다는 것은 마우스를 눌렀을 때(mouse down)의 위치를 계산하고, 마우스를 띄었을 때(mouse up)의 위치를 계산해서 회색으로 바탕을 변경해 주는 기능을 한다. 인터넷 사이트를 만들 때 블럭이 가능하다는 것은 복사가 용이하다는 것인데 이것보다 더 불편한 것은 사용자가 화면에서 마우스를 클릭했을 때 화면의 이미지가 변경되는 불편함에 있다. 그래서 마우스를 클릭했을 때 블럭처리를 하지 않기 위해서 만드는 가장 간단한 방법은 마우스를 눌렀을 때 아무런 동작도 하지 않도록 처리하는 것이다. 마우스를 눌렀을 때 발생하는 이벤트는 'onmousedown'이고 마우스를 띄었을 때 발생하는 이벤트는 'onmouseup'인데, 여기서는 'onmousedown' 이벤트에서 아무런 동작도 하지 않도록 처리해서 브라우저에서의 블럭처리를 금지할 수 있다. 그래서, [코드 4-3]과 같이 브라우저 전체에서 마우스를 눌렀을 때 아무런 동작을 하지 않도록 처리했으며 [그림 4-3]은 그 결과를 보여준다. [코드 4-3]에서 jQuery를 사용해서 브라우저 전체에서 마우스를 동작하지 않게 하기 위해서 $(window)를 사용해서 윈도우 전체 객체를 가져와서 bind() 함수로 마우스 왼쪽 버튼과 오른쪽 버튼이 클릭되었을 때의 이벤트를 처리하는 코드를 추가하였다. 여기서 주의 깊게 봐야할 이벤트 이름은 jQuery에서는 마우스 왼쪽 버튼이 클릭되었을 때 발생하는 이벤트 이름을 'mousedown'으로 사용하고, 마우스 우측 버튼이 클릭되었을 때 브라우저의 메뉴가 생성되는 이벤트 이름을 'contextmenu'로 사용한다. 이 책의 부록에서 설명하는 브라우저의 개발자 모드로 웹 프로그램의 모든 소스코드를 확인할 수 있지만, 일반 사용자는 마우스 우측 버튼을 눌러서 웹 블로그의 내용을 파일로 저장하거나 복사하는 것을 이와 같은 방법으로 막을 수 있다. 인터넷에서는 정보의 공유를 원칙으로 하지만, 그래도 긴 시간동안 소중히 작성한 내용을 누군가가 쉽게 복사해서 가져간다는 것은 많은 블러거들이 좋아하지 않는 편이다.

확인 사이트: http://jisiksoft.com/book/4/chapter_4/4.1/2/www/

그림 4-3 화면에서 마우스 드래그에도 텍스트에 블럭이 만들어지지 않는다.



[코드 4-3] 화면에서 마우스 드래그에도 블럭이 만들어지지 않는 방법 (/2/www/index.html)

```
<!DOCTYPE html>

<head>
    <title>MiddleStory</title>
    <meta charset="utf-8"></meta>
     <meta name="keywords" content="미들스토리, middlestory, middle, story"></meta>
     <link rel="stylesheet" href="./css/jquery-ui.min.css"></link>                    //---①
    <link rel="stylesheet" href="./css/middlestory.css"></link>
    <script src="./js/jquery-2.1.3.min.js"></script>
    <script src="./js/jquery-ui.min.js"></script>
    <link rel="stylesheet" href="./css/middlestory.css"></link>
</head>
<body>
    <div id="header">This is Header.</div>
     <div id="container">This is Container.</div>
    <div id="footer">This is Footer.</div>

    <script>
     $(window).bind("mousedown contextmenu", function() {                    //---②
        return false;
     });
    </script>
</body>
</html>
```

① jQuery를 사용하기 위해서는 'jquery.min.js' 파일만 연결해서 사용하면 되지만, 일반적으로 jQuery의 많은 기능은 'jquery-ui.min.js' 파일에도 많이 만들어져 있다. 그래서, jQuery를 사용할 때는 'jquery-ui.min.css', 'jquery-ui.min.js', 'jquery.min.js' 세 개의 파일을 모두 연결해서 사용하는 것이 좋다.

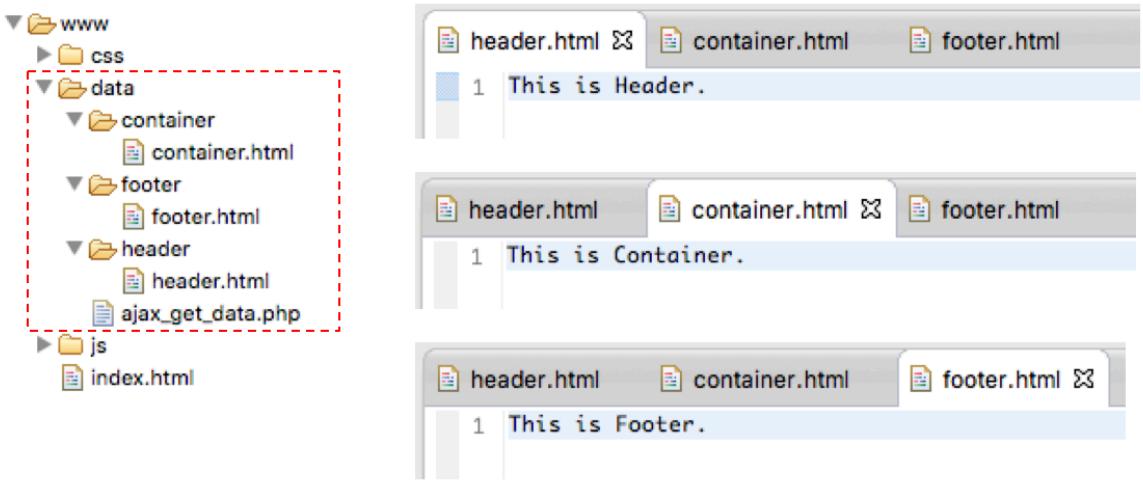
② 마우스를 클릭했을 때 브라우저에서 아무런 동작도 하지 않기 위해서 jQuery를 $(window)와 같이 사용해서 브라우저 전체의 객체로 접근해서 이벤트를 연결시키는 bind() 함수를 사용한다. 마우스 왼쪽 버튼을 클릭했을 때 발생하는 'mousedown' 이벤트와 마우스 오른쪽 버튼을 눌렀을 때 열리는 브라우저의 메뉴를 여는 'contextmenu' 이벤트가 발생했을 때 아무런 동작도 하지 않게 하기 위해서 'return false;'로 처리했다.

웹 블로그는 인터넷 상에서 정보를 제공하는 역할을 하기 때문에 사용자가 입력해야 하는 부분이 없는 경우가 많다. 이와 같이 마우스를 클릭했을 때 아무런 동작을 하지 않게 하는 것이 좋은데, 때때로 사용자가 내용을 입력해야 하는 <input> 태그

를 만든다면 여기서 사용한 'mousedown' 이벤트를 사용해서는 안된다. 'mousedown' 이벤트 발생 시 'return false;'로 아무런 동작을 하지 않게 하면 텍스트를 입력하는 <input> 태그를 선택하는 것도 불가능하기 때문이다. <input> 태그를 선택하는 것이 'click' 이벤트를 사용하면 좋지만, 현재는 대부분의 브라우저가 'mousedown' 이벤트에서 처리되도록 만들어졌다.

브라우저는 서버로부터 데이터를 가져와서 해당 위치에 데이터를 보여주는 역할을 하는데, [그림 4-4]는 각 화면의 데이터를 가져오기 위해서 '/data/'라는 디렉토리를 만들고 디렉토리 안에 분할된 화면의 데이터를 구분하기 위해서 '/header/', '/container/', '/footer/' 라는 세 개의 디렉토리를 추가했다. 또한 각각의 디렉토리에는 '.html' 파일들이 존재하는데 [그림 4-4]의 오른쪽과 같이 텍스트로 파일의 내용만을 표기하였다. 이번에 만드는 코드는 이와 같이 분할된 영역의 데이터를 가져와서 브라우저에 넣어주는 방법을 사용하는데, 데이터를 가져오기 위해서 Ajax를 사용하고 서버에서는 PHP를 사용해서 해당 위치의 파일을 읽어서 내용을 브라우저로 보내주는 방법을 구현한다.

그림 4-4 분할된 화면의 영역의 내용을 서버로부터 받아서 보여주기 위한 /data/ 디렉토리 내용



[코드 4-4]부터 [코드 4-6]까지의 내용은 서버로부터 데이터를 가져와서 화면에 넣어주는 코드를 보여주는데, 앞으로 계속 구현되는 사이트의 기본 원리는 여기서 출발한다고 이해하면 된다. 즉, 불러오는 데이터를 적절히 변경해주면서 브라우저의 내용을 변경할 수 있다. [코드 4-4]의 HTML 코드는 간단한데, 이후에 많은 코드들이 추가되지만 'index.html' 파일은 변경 없이 블로그가 만들어진다. 또한, Ajax를 사용해서 해당내용을 가져와서 브라우저에 추가하는 방법을 사용하게 되면, 브라우저의

주소 창에 나타나는 http://middlestory.com 도메인 이름은 아무런 변화가 없기 때문에 복잡한 주소를 보여주기도 하는 다른 사이트에 비해서 보기가 좋다.

[코드 4-4] 서버로부터 데이터를 가져오는 방법을 위한 HTML 파일 (/3/www/index.html)

```
<!DOCTYPE html>

<head>
    <title>MiddleStory</title>
    <meta charset="utf-8"></meta>
     <meta name="keywords" content="미들스토리, middlestory, middle, story"></meta>
     <link rel="stylesheet" href="./css/jquery-ui.min.css"></link>
    <link rel="stylesheet" href="./css/middlestory.css"></link>
    <script src="./js/jquery-2.1.3.min.js"></script>
    <script src="./js/jquery-ui.min.js"></script>
    <script src="./js/middlestory.js"></script>
</head>
<body>
    <div id="header"></div>
     <div id="container"></div>
    <div id="footer"></div>

    <script>
     $(window).bind("mousedown contextmenu", function() {
        return false;
     });

     initialize();                                                    //---①
    </script>
</body>
</html>
```

① <body></body> 태그 안에는 세 개의 태그들이 있으며, 아무런 내용도 가지고 있지 않다. 브라우저에 코드가 받아지면 initialize()' 함수가 자동으로 실행되고 서버로부터 데이터를 가져와서 세 개의 태그들 안에 내용을 넣어준다.

[코드 4-5] 서버로부터 데이터를 가져오는 JavaScript 파일 (/3/www/middlestory.js)

```
function initialize() {

    var param = { dataId:"header" };                                 //---①
    var result = doAjax('./data/ajax_get_data.php', param);
    $("#header").empty().html(result);
```

```
    var param = { dataId:"container" };                                    //---②
    var result = doAjax('./data/ajax_get_data.php', param);
    $("#container").empty().html(result);

    var param = { dataId:"footer" };
    var result = doAjax('./data/ajax_get_data.php', param);                //---③
    $("#footer").empty().html(result);
}

function doAjax(strUrl, inputData) {

    var result;

    $.ajax({
            url: strUrl,
            type: 'post',
            async: false,
            datatype: 'json',
            data: inputData,
            error: function() {
                    alert('Use Chrome or FireFox instead of Explorer.');
            },
            success: function(obj) {
            result = obj;
            }
            });
    return result;
}
```

① 화면의 'header' 영역의 데이터를 가져오기 위해서 전달되는 'dataId' 값에 'header'라는 이름을 넣어주었다. 여기서 호출하는 함수는 서버의 'ajax_get_data.php' 파일이며, 결과를 받아서 'header'라는 id 값을 가지고 있는 태그 안에 넣어준다.

② 화면의 'container' 영역의 데이터를 가져와서 'container'라는 id 값을 가지고 있는 태그 안에 넣어준다.

③ 화면의 'footer' 영역의 데이터를 가져와서 'footer'라는 id 값을 가지고 있는 태그 안에 넣어준다.

[코드 4-6] 데이터 요청에 해당 파일의 내용을 보내는 PHP 코드 (/3/www/data/ajax_get_data.php)

```
<?php

$dataId          = (isset($_POST['dataId'])) ? $_POST['dataId'] : "";

if ($dataId == "header")                                                   //---①
```

```
        $filename = "./header/".$dataId.".html";
else if ($dataId == "footer")
        $filename = "./footer/".$dataId.".html";
else
        $filename = "./container/".$dataId.".html";

$fp = fopen($filename, "r");                                                    //---②
$data = fread($fp, filesize($filename));
fclose($fp);

print $data;
exit();

?>
```

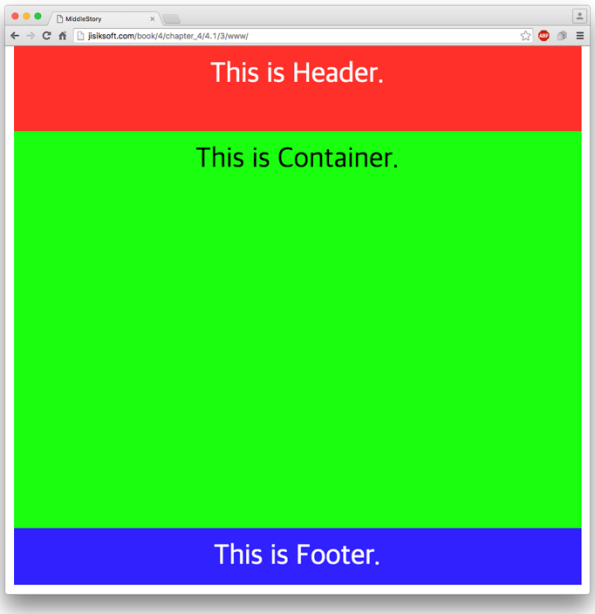
① 'Post 방식'으로 받는 'dataId' 값에 기반해서 읽기 위한 파일의 이름을 정한다. 여기서는 디렉토리로 구분되었기 때문에 해당 디렉토리를 포함해서 파일 이름을 갖게 된다.

② 해당 파일을 읽어서 내용을 브라우저로 전달한다.

[그림 4-5]는 이전과 같은 결과를 보여주는데, 분할된 각각의 영역은 매번 서버로부터 데이터를 가져와서 결과를 해당 위치에 넣어준 것이다. 실제 블로그에서는 'Header'와 'Footer' 영역의 데이터는 사이트에 접속하면 한 번만 가져오게 되며, 사용자가 브라우저를 보는 동안에는 'Container' 영역의 데이터를 매번 가져와서 내용을 변경해주는 방법으로 진행된다. 오래전에는 브라우저의 전체 화면을 새로 받아서 브라우저에 보여주는 방법으로 사이트들이 만들어졌지만, Ajax 기술이 만들어진 이후에는 변경이 필요한 영역의 데이터만 가져와서 내용을 변경해주는 방법으로 진행된다. 쉽게 이해하기 위해서는 처음 방문할 때 주소 창에 도메인을 가지고 방문했어도 사이트에 머무는 동안에도 도메인 이름이 변경되는 경우가 발생하는데, 이런 경우에는 새로운 페이지를 다시 가져온 것이고, 도메인 이름이 변경되지 않는다면 Ajax로 현재의 페이지에서 내용만 변경한 것으로 이해하면 된다. 필자는 Ajax로 데이터를 가져와서 특정 영역의 데이터만 변경해주는 방법을 좋아하는데, 브라우저의 주소 창의 도메인이 변경되지 않는 것이 깔끔하다고 생각하기 때문이다. 물론, 대형 사이트에서는 여러 페이지들이 독립적으로 구성되기 때문에 어쩔 수 없지만, 블로그나 작은 회사의 사이트에서는 특별히 독립된 여러 페이지들을 만들 필요가 없다고 볼 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_4/4.1/3/www/

그림 4-5 서버로부터 데이터를 가져와서 분할된 화면에 넣어준 결과



사이트의 특정 파일의 내용을 읽어서 브라우저로 보내주는 방법을 이해하면, 웹 서버를 만들기 위해서 설치하는 '아파치'(Apache) 소프트웨어를 이해할 수 있다. '아파치' 소프트웨어를 간단히 생각하면 특정 영역의 파일을 읽어서 HTML 프로토콜 방식으로 파일의 내용을 브라우저로 보내주는 기능을 한다. 그래서 '아파치'를 설치하면 파일을 읽기 위한 기본 위치를 변경할 수도 있으며, 한 서버에서 다수의 도메인 사이트를 만든다는 것은 특정 도메인에 대한 요청이 올 때 해당 도메인과 연결된 서버의 특정 디렉토리의 파일을 읽어서 보내주도록 설정했다는 것이다. 소프트웨어의 기능을 단순화해서 생각하면 많은 것들이 이해되니, 세세한 부분을 너무 기억하려고 하다가 지치지 않길 바란다.

4.2 메뉴 만들기

이번 장에서는 블로그의 상단에 있는 메뉴를 만드는데, [그림 4-6]은 분할된 화면에서 상단(Header 영역)과 하단(Footer 영역)에 내용이 입력된 결과를 보여준다. 브라우저에서 보여지는 이미지는 이미지파일을 알맞은 위치에 배치한 것이며, 상단의 메뉴가 클릭되면 중간(Container 영역)에 블로그 내용을 서버로부터 가져와서 변경한

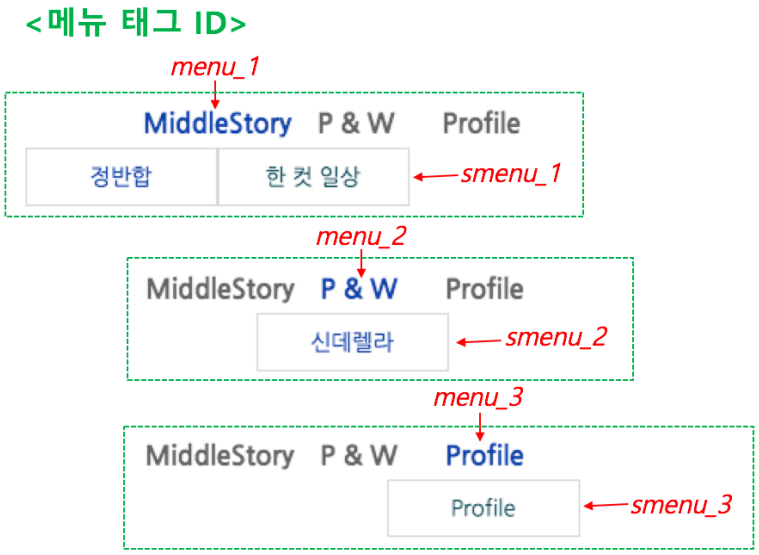
다. 블로그 페이지가 열리면 상단과 하단의 내용은 변경되지 않으며, 중간의 내용만 변경되기 때문에 이번 장에서는 가장 중요한 메뉴를 만드는 방법을 설명한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_4/4.2/www/

그림 4-6 메뉴를 선택했을 때 Container 에 해당 메뉴의 내용이 보인다.

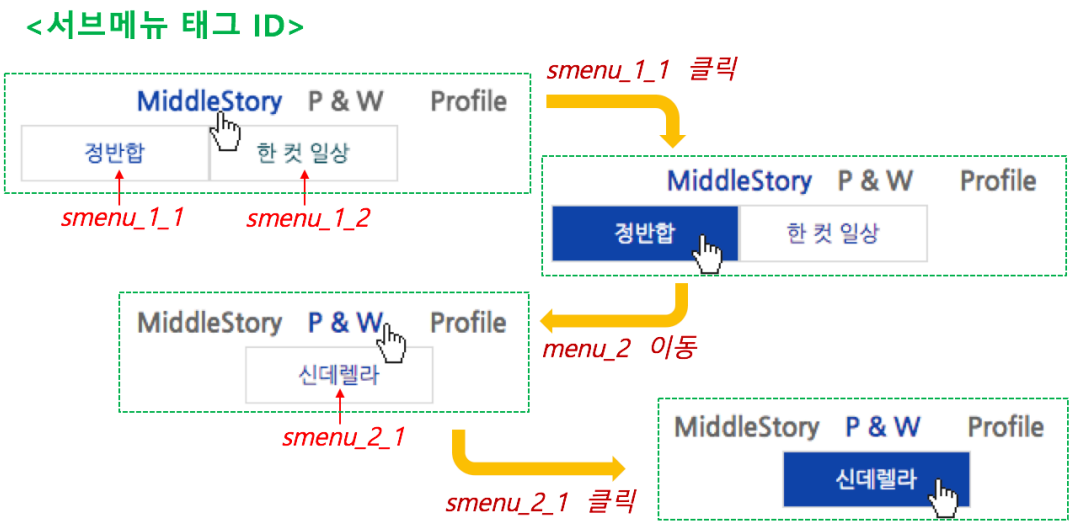
이 책에서 사용하는 태그는 대부분 태그만을 사용하는데, 태그가 객체이며 고유한 id 값을 갖는다는 것을 이해하면 메뉴가 클릭될 때마다 화면에 보여지는 객체를 선택한다는 것을 이해할 수 있을 것이다. 즉, 각각의 메뉴는 고유 id 값을 가지고 있으며, 해당 메뉴가 클릭되면 보여주어야 하는 id 값을 갖는 객체를 화면에 나타내고 필요 없는 객체는 보이지 않게 처리하면 된다. [그림 4-7]은 이러한 과정을 보여주는데, 'menu_1'(Menu)이라는 id 값을 갖는 메뉴가 선택되면 'smenu_1'(Sub-Menu)라는 id 값을 갖는 객체를 화면에 보여준다. 이와 같은 방법으로 고유의 id 값을 가지고 자신의 하위 메뉴(Sub-Menu)를 화면에 보여주고 나머지 하위 메뉴들은 화면에서 보이지 않게 처리하면 된다.

그림 4-7 각각의 메뉴는 고유 id 값을 가지고 있다.



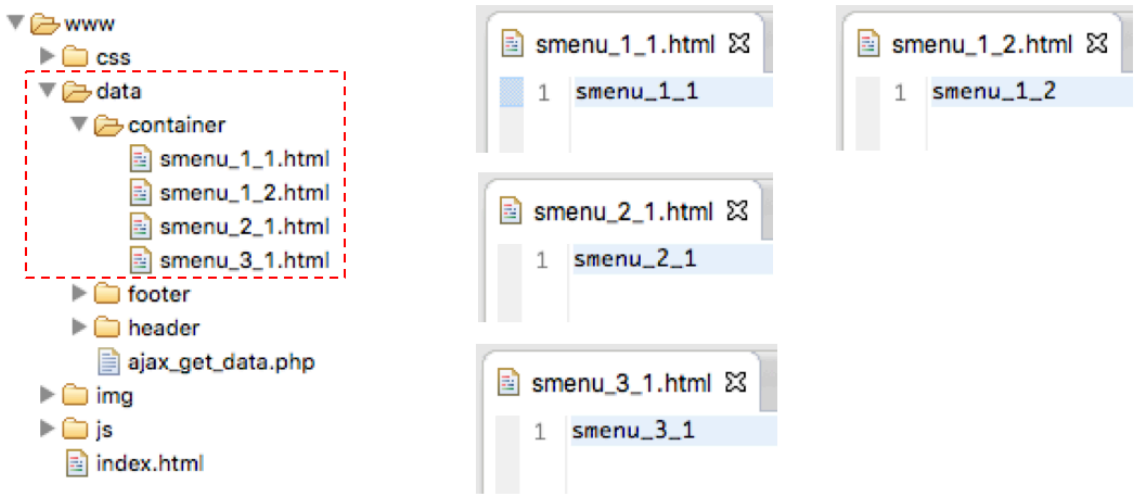
[그림 4-8]은 메뉴의 하위 메뉴 안에 있는 항목들을 클릭할 경우에 처리되는 과정을 보여주는데, id 값은 일정한 규칙을 따른다. 예를 들면, [그림 4-7]에서 'MiddleStory'는 'menu_1'이라는 id 값을 갖고, 'menu_1'의 하위 메뉴Sub-Menu는 'smenu_1'이라는 id 값을 갖는다. 그리고, [그림 4-8]과 같이 'smenu_1'이라는 객체 안에는 두 개의 항목이 있는데, 각각을 'smenu_1_1'과 'smenu_1_2'라는 id 값을 부여하였다. 이와 같은 방법으로 하위 메뉴에 다수의 선택할 수 있는 항목이 추가되어도 프로그래머가 정한 규칙에 따른 고유 id 값을 부여하면서 메뉴의 틀을 만들어주면 된다. [그림 4-8]에서와 같이 하위 메뉴를 선택했을 때 선택된 항목의 글자색과 바탕색을 변경해주는 처리를 해서 어떤 항목이 클릭되었는지를 사용자는 알수 있다.

그림 4-8 메뉴에 마우스가 옮겨지거나 클릭될 때는 태그의 id 를 기준으로 동작한다.



[그림 4-8]에서 하위 메뉴의 선택항목이 클릭되면, 가운데의 Container 영역의 내용을 변경해주면 되는데, [그림 4-9]는 Container 안에 들어갈 내용들이 해당 메뉴의 id 값과 같은 이름의 파일을 가지고 있음을 알 수가 있다. 즉, [그림 4-8]에서 'smenu_1_1'이라는 태그 객체가 클릭되면, 서버로부터 'smenu_1_1.html'이라는 파일의 내용을 가져와서 Container 영역에 출력시킨다. 이와 같은 방법으로, 필요한 메뉴의 id 값을 정하고 해당 id 값을 파일이름으로 만들어서 관리하면 체계적인 메뉴와 파일 관리가 쉽다. 웹 프로그래밍에서 태그의 id 값을 정하는 것이 아주 중요한데, 사이트를 구축할 때 id 값으로 태그의 성격을 알 수 있도록 하는 것은 상당히 중요하며 이후에 유지보수 하는데도 도움이 많이 된다.
[그림 4-9]의 오른쪽에는 해당 파일들의 내용을 보여주는데, 텍스트로 파일의 이름이 있으며 메뉴의 버튼이 클릭될 때마다 Container 영역에 보여지는 내용이다.

그림 4-9 메뉴가 클릭되었을 때 Container 영역으로 가져오는 SubMenu 의 이름과 같은 파일들



브라우저에서 http://middlestory.com에 접속하면 자동으로 가져오는 'index.html' 파일에는 변경이 없으며 분할된 영역의 파일들을 자동으로 가져와서 화면에 보여준다. [코드 4-7]은 메뉴를 만들기 위하여 상단(Header 영역)에 위치하는 'header.html' 파일의 코드이며, 태그들만 사용해서 클래스와 id 값들을 갖고 디자인과 이벤트가 처리된다. 각각의 태그 안에서는 다양한 이벤트가 처리되는데, 마우스가 태그 위에 위치할 때 발생하는 'onmouseover', 태그 밖으로 마우스가 이동할 때 발생하는 'onmouseout', 태그가 클릭되었을 때 발생하는 'onclick' 이벤트 들을 사용하였다. 코드에서 주의 깊게 봐야할 것 중의 하나가 함수의 매개변수로 'this.id'를 넣

어주는데, this라는 것은 현재 자신의 객체를 의미하기 때문에 this가 사용되는 태그 객체를 의미하며 id라는 값을 사용해서 'this.id'는 태그 객체의 id 값을 의미이다. [코드 4-8]은 디자인을 위한 CSS가 설정되었으며, 태그들의 위치와 다양한 디자인들을 위해서 만들었다. [코드 4-9]는 이벤트가 발생할 때 처리되는 JavaScript 코드이며, 메뉴를 처리하는 함수들을 주의 깊게 봐야할 부분은 id 값을 갖는 객체들을 'display' 속성을 사용해서 화면에 보여주고 숨기는 객체들을 관리한다는 것이다. 즉, 'display:none'이라고 설정하면 해당 객체가 화면에서 보이지 않으며, 'display:block'이라고 하면 해당 객체가 화면에 나타난다.

[코드 4-7] 블로그 상단에 위치하는 메뉴의 HTML 코드 (/data/header/header.html)

```
<div id="logo_middlestory" onclick="clickedSubMenu('smenu_1_1')">                    //---①
      <img src="./img/common/middlestory_1.png">
</div>
<div id="menu_area">                                                                 //---②
      <div class="menu" id="menu_1" onmouseover="mouseOverMenu(this.id);"            //---③
         onclick="clickedMenu(this.id);">MiddleStory</div>
      <div class="menu" id="menu_2" onmouseover="mouseOverMenu(this.id);"
         onclick="clickedMenu(this.id);">P & W</div>
      <div class="menu" id="menu_3" onmouseover="mouseOverMenu(this.id);"
         onclick="clickedMenu(this.id);">Profile</div>
</div>

<div class="smenu_area" id="smenu_1">                                                //---④
      <div class="smenu" id="smenu_1_1"                                              //---⑤
         onmouseover="mouseOverSubMenu(this.id);"
         onmouseout="mouseOutSubMenu(this.id);"
         onclick="clickedSubMenu(this.id);">정반합</div>
      <div class="smenu" id="smenu_1_2"
         onmouseover="mouseOverSubMenu(this.id);"
         onmouseout="mouseOutSubMenu(this.id);"
         onclick="clickedSubMenu(this.id);">한 컷 일상</div>
</div>

<div class="smenu_area" id="smenu_2" style="display: none;">
      <div class="smenu" id="smenu_2_1"
         onmouseover="mouseOverSubMenu(this.id);"
         onmouseout="mouseOutSubMenu(this.id);"
         onclick="clickedSubMenu(this.id);">신데렐라</div>
</div>
```

```
<div class="smenu_area" id="smenu_3" style="display: none;">
      <div class="smenu" id="smenu_3_1"
         onmouseover="mouseOverSubMenu(this.id);"
         onmouseout="mouseOutSubMenu(this.id);"
         onclick="clickedSubMenu(this.id);">Profile</div>
</div>
```

① 홈페이지에서 이미지를 보여주기 위한 태그이며, 이미지를 클릭하면 'smenu_1_1.html' 파일을 화면에 보여준다.

② 상단의 메뉴 항목들을 감싸기 위해서 'menu_area'라는 id 값을 가진 태그를 사용했으며, 화면에서 메뉴의 위치와 디자인을 위해서 사용했다. 이와 같이, 태그들을 감싸는 태그를 만들어서 사용하는 방법은 아주 요긴하게 쓰인다.

③ 각각의 메뉴들은 디자인을 위해서 'menu'라는 클래스 값을 가지고, 이벤트 처리를 위해서 고유한 id 값을 가진다.

④ 메뉴의 하위 메뉴들은 공통된 디자인을 위해서 'smenu_area'라는 클래스 값을 가지며, 하위 메뉴의 id 값은 화면에서의 위치를 정하기 위해서 사용된다.

⑤ 하위 메뉴에서 선택하는 항목들은 모두 태그들을 사용해서 만들어지며, 이벤트 처리를 위해서 고유한 id 값을 갖는다.

[코드 4-8] 메뉴의 위치를 포함한 태그 객체들의 디자인을 위한 CSS 코드 (/css/middlestory.css)

```
@import url(http://fonts.googleapis.com/earlyaccess/nanumgothic.css);          //---①

body {                                                                          //---②
      width: 1000px;
      margin: auto;
      color: #0d4a57;
      font-family: 'Nanum Gothic', sans-serif;
      font-size: 17px;
      line-height: 35px;
      padding: 0;
}
img {                                                                           //---③
       height:100%;
       width:100%;
}
#logo_middlestory {
      position: absolute;
      left: 50px;
      top: 10px;
      width: 150px;
```

```
}

#header {
        position: relative;
        height: 150px;
}
#container {
         position: relative;
        text-align: center;
         min-height: 700px;
         font-size: 60px;
         line-height: 220px;
         color: #000000;
        background-color: #33ff33;
}
#footer {
        position: relative;
        text-align: left;
        font-size:14px;
        line-height: 20px;
        color: #1646a7;
}

#menu_area {                                                    //---④
        position: absolute;
        top: 30px;
        left: 575px;
        width: 350px;
        height: 100px;
        margin: 30px 0 0;
        color: #666666;
}
.menu {                                                         //---⑤
         width: 80px;
         height:40px;
        font-weight: bold;
        text-align: right;
        list-style: outside none none;
        float:left;
}
.smenu_area {                                                   //---⑥
         position: absolute;
         top: 63px;
         width: 425px;
         height: 100px;
         margin: 30px 0 0;
}
#smenu_1 {                                                      //---⑦
         left: 500px;
```

```
}
#smenu_2 {
        left: 645px;
}
#smenu_3 {
       left: 728px;
}
.smenu {                                                                    //---⑧
        position:relative;
       width: 120px;
       height: 35px;
       border: 1px solid #dadada;
       font-size: 14px;
       line-height: 35px;
       text-align: center;
       cursor: pointer;
       float:left;
}

.line_hori {                                                                //---⑨
        height:6px;
        border-top:    1px solid #1646a7;
        border-bottom: 0px;
        border-left: 0px;
        border-right: 0px;
}
```

① 브라우저에서 보여지는 글자체는 이미 설치된 글자체를 사용하는데, 디자인에서 설정하여도 모든 브라우저에서 같은 글자체를 보여주지 않는 경우가 많다. 그래서 모든 브라우저에서 같은 글자체를 사용하기 위해서 페이지가 열릴 때 글자체를 자동으로 설치하게 만들어야 하는데, 여기서는 가장 많이 사용하는 '나눔고딕'을 다운받아서 설치하게 된다.

② 브라우저 전체에서 공통적으로 사용하는 디자인을 설정하는데, 대표적인 것이 내용이 출력되는 화면의 가로 길이와 글자체를 '나눔고딕'으로 설정하였다.

③ 이미지는 가로와 세로의 비율을 100%로 정하였는데, <img> 태그를 태그로 감싸서 사각형의 위치와 크기를 정하면 이미지가 태그 안에서 꽉 채워져서 출력된다.

④ 메뉴 전체의 위치를 포함한 디자인을 설정한다.

⑤ 각 메뉴의 크기와 디자인을 설정한다.

⑥ 다수의 하위 메뉴들의 위치와 디자인을 설정한다.

⑦ 각각의 하위 메뉴들의 위치가 다르기 때문에, id 값을 사용해서 개별적으로 위치를 잡아주었다.

⑧ 하위 메뉴에 있는 각각의 선택 항목들에 대한 크기를 포함한 디자인들을 설정했다.

⑨ 화면의 아래쪽(Footer 영역)에 구분하기 위한 선을 한 줄 그리기 위해서, 수평선 Horizon Line을 의미하는 'line_hori'라는 클래스 값을 사용해서 디자인을 설정했다. 이후에 가로선을 그리기 위해서는 HTML 코드에서 ' '를 추가하면 된다.

[코드 4-9] 메뉴위로 마우스를 이동하거나 클릭했을 때 동작하는 JavaScript 코드 (/js/middlestory.js)

```
var currMenu = "menu_1";                                                    //---①
var currSubMenu = "smenu_1_1";                                              //---②
var currPage = "smenu_1_1";                                                 //---③

function mouseOverMenu(mId) {                                               //---④
    $("#"+currMenu).css({ "color": "#666666" });
    $("#s"+currMenu).css({ "display": "none" });
    $("#"+mId).css({ "color": "#1646a7" });
    $("#s"+mId).css({ "display": "block" });

    currMenu = mId;
}

function clickedMenu(mId) {                                                 //---⑤
    clickedSubMenu('s'+mId+'_1');
}

function mouseOverSubMenu(smId) {                                           //---⑥
    $("#"+smId).css({ "color": "#ffffff", "background-color": "#1646a7" });

    currSubMenu = smId;
}

function mouseOutSubMenu(smId) {                                            //---⑦
    if (currPage != currSubMenu) {
        $("#"+smId).css({ "color": "#1646a7", "background-color": "#ffffff" });
    }
}

function clickedSubMenu(smId) {                                             //---⑧
    $("#"+currPage).css({ "color": "#1646a7", "background-color": "#ffffff" });
    $("#"+smId).css({ "color": "#ffffff", "background-color": "#1646a7" });
```

```
    currPage = smId;

    var param = { dataId:smId };
    var result = doAjax('./data/ajax_get_data.php', param);
    $("#container").empty().html(result);
}

function initialize() {                                                    //---⑨
    var param = { dataId:"header" };
    var result = doAjax('./data/ajax_get_data.php', param);
    $("#header").empty().html(result);

    mouseOverMenu("menu_1");
    clickedSubMenu("smenu_1_1");

    var param = { dataId:"footer" };
    var result = doAjax('./data/ajax_get_data.php', param);
    $("#footer").empty().html(result);
}

function doAjax(strUrl, inputData) {

    var result;

    $.ajax({
            url: strUrl,
            type: 'post',
            async: false,
            datatype: 'json',
            data: inputData,
            error: function() {
                    alert('Use Chrome or FireFox instead of Explorer.');
            },
            success: function(obj) {
            result = obj;
            }
            });
    return result;
}
```

① 마우스가 가리키는 메뉴Current Menu를 확인하기 위해서 currMenu 변수를 사용하며, 초기값으로 첫 번째 메뉴의 id 값을 저장한다.

② 마우스가 가리키는 하위 메뉴Current SubMenu를 확인하기 위해서 currSubMenu 변수를 사용하며, 초기값으로 첫 번째 하위 메뉴의 id 값을 저장한다.

③ 블로그의 내용이 보여지는 'Container' 영역에서 보여지는 내용을 알기 위해서 currPage 변수를 사용하며, 마우스를 이동할 때 하위 메뉴의 디자인 변경을 위해서 사용한다.

④ 마우스가 메뉴 위로 옮겨질 때 실행되는 함수로서, 활성화된 메뉴를 회색으로 변경하고 기존의 하위 메뉴를 숨긴다. 이후에, 선택된 메뉴를 활성화된 디자인으로 변경하고 해당 메뉴의 하위메뉴를 화면에 보여준다. 또한, 현재 활성화된 메뉴가 변경되었기 때문에 currMenu 변수의 값을 변경한다.

⑤ 메뉴가 클릭되면 실행되는 함수로서 하위 메뉴의 첫 번째 항목을 클릭한 결과를 출력한다.

⑥ 마우스가 하위 메뉴 위로 옮겨질 때 실행되는 함수로서 선택된 하위 메뉴 항목의 글자색과 바탕색을 활성화된 상태로 변경한다.

⑦ 마우스가 하위 메뉴 밖으로 옮겨질 때 실행되는 함수로서 하위 메뉴가 현재 'Container' 영역에 출력된 내용이 아니면 비활성화된 상태로 변경한다.

⑧ 하위 메뉴가 클릭되었을 때 실행되는 함수로서 이미 열려진 페이지의 하위 메뉴를 비활성화시키고, 현재 클릭된 하위 메뉴를 활성화된 이미지로 변경한다. 이후에 서버로부터 하위 메뉴와 같은 이름을 가진 파일내용을 받아서 'Container' 영역에 넣어준다.

⑨ 브라우저에서 블로그가 열릴 때 자동으로 시작되는 함수가 initialize() 함수이며, 'Header', 'Container', 'Footer' 내용들을 서버로부터 가져와서 브라우저에 출력한다.

분할된 영역의 항목들을 파일에서 관리하는 것은 상당히 편리한 방법이며, 여러사람들이 모여서 분할된 페이지들을 만들고 디자인할 때 개인별로 파일을 맡아서 관리할 수 있기 때문에 편리하다. [그림 4-10]은 화면의 하단(Footer 영역)에 넣기 위한 HTML 코드이며, 두 개의 그림을 넣고 수평선을 그리고 난 이후에 블로그 정보를 기입하였다.

[코드 4-10] 블로그의 아래에 항상 보여지는 내용을 위한 HTML 코드 (/footer/footer.html)

```
<div style="position:relative;height:100px;float:right;">                    //---①
      <img src="./img/common/middlestory_2.png">
</div>
<div style="position:relative;top:38px;right:50px;height:60px;float:right;">
      <img src="./img/common/middlestory_3.png">
</div>

<div class="line_hori" style="clear:right;"></div>                           //---②
  Email : hjnam@jisiksoft.com </br>
  Copyright (C) MIDDLESTORY All Rights Reserved</br>
</br></br></br></br></br>
```

① 그림을 오른쪽에 넣기 위해서 'float' 속성을 right으로 설정하였다.
② 수평선으로 그림과 문자를 구분하기 위해서 'line_hori'라는 클래스 값을 가지고 있는 태그를 사용했다. 또한 이전의 태그에서 'float:right'을 사용했기 때문에, 'right'이라는 속성값을 지우기 위해서 'clear:right'을 사용하였다.

브라우저에서 보여지는 사이트들은 짧은 시간에 만들기가 힘들다. 하나의 디자인을 위해서 많은 작업들을 진행해야 하기 때문이다. 서버의 내용도 만들어야 하지만 브라우저에서 보여지는 페이지만을 만드는 것도 상당한 시간이 필요하다. 물론, 브라우저에서 보여지는 내용을 만드는 것이 가장 중요하지만, 디자인 자체만으로도 화면의 픽셀 위치를 맞추기 위해서 많은 단순 작업이 필요하다. 웹 프로그래밍도 다른 프로그래밍과 마찬가지로 조급한 마음보다는 화면에 내용을 변경하면서 느끼는 즐거움을 가지고 만드는 것이 좋다고 생각한다.

4.3 내용 불러오기

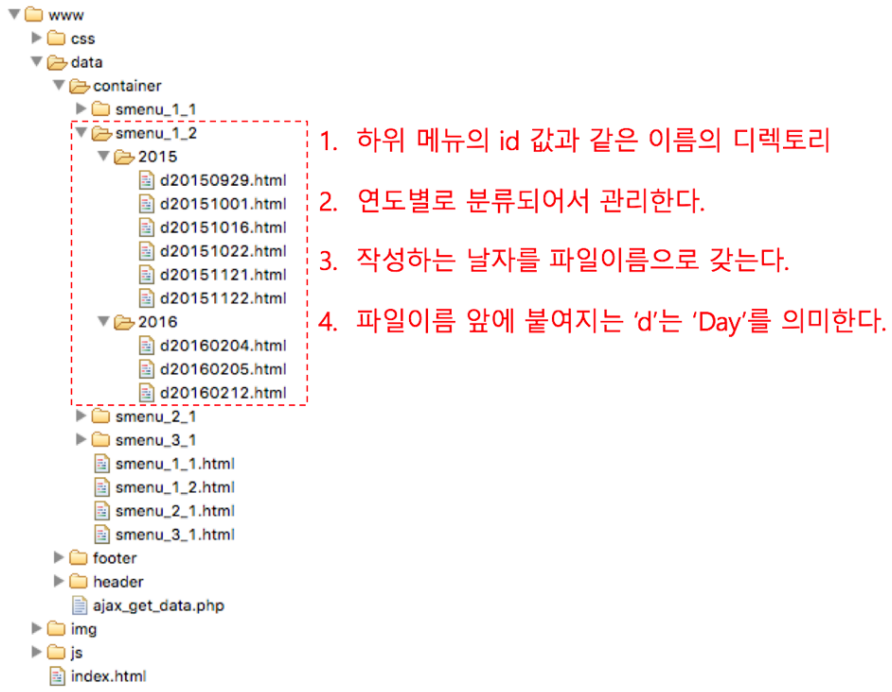
이전 장까지의 내용으로 블로그의 골격은 모두 갖추어졌다. 이제부터는 블로그의 내용을 위해서 Container 영역을 만들 차례이다. 웹 프로그래밍에서 중요한 것 중의 하나는 id의 관리이다. 블로그의 내용은 서버에 저장된 파일을 읽어서 화면에 보여주는 것인데, 블로그의 특성 상 많은 글들을 작성하기 때문에 파일의 갯수가 많아질 수밖에 없다. 그래서, 관리해야하는 파일의 이름을 분류하기 쉽게 해야 하는데, 파일 이름을 날짜로 하는 방법이 가장 편하다. 오랫동안 블로그를 사용한다면 연도 별로도 파일들을 관리할 수 있다. 이번 장에서는 파일들을 어떻게 관리하는지를 설명한다. 물론 독자들은 필자의 방법을 하나의 예제로 받아들이고 자신만의 방법으로 파일 관리하는 방법을 찾아가길 바란다.
[그림 4-10]은 블로그의 내용이 들어간 것을 보여주는데, Container 영역의 왼쪽에 블로그를 선택할 수 있는 메뉴가 만들어졌다. 블로그 상단의 Header 영역은 넓은 범위의 분류를 위한 것이고, Container 영역의 메뉴는 특정 주제로 구분된 항목의 블로그 내용들을 선택할 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_4/4.3/www/

그림 4-10 블로그의 내용이 Container 영역에 보여지며 왼쪽에 새로운 메뉴가 존재한다.

[그림 4-11]은 서버에 저장된 파일들을 보여주는데, 블로그를 작성한 날짜의 이름으로 파일이 관리된다. 또한, 블로그의 특성상 만들어지는 파일이 많기 때문에 연도별로 파일을 나누었으며, 만약 아주 많은 파일들을 만들어야 한다면 월 별로 관리하는 것도 하나의 방법이다. 연도 별로 분류된 항목에 접근하는 방법을 이해한 독자라면 월 별 방법도 쉽게 구현이 가능할 것이다. [그림 4-11]에서 블로그 상단의 하위 메뉴가 클릭되었을 때 Container 영역에 넣는 파일들은 'smenu_1_2.html'과 같은 형태이며, 'smenu_1_2.html' 파일 안에서 메뉴를 만들어서 'smenu_1_2'와 같이 같은 이름을 갖는 디렉토리에서 파일을 가져온다. 날짜로 관리되는 파일들은 앞에 'd'라는 문자가 항상 있는데 'day'의 첫 글자이다. 블로그의 Container 영역의 메뉴를 관리할 때 id 값을 정하는 규칙이 있는데, 연도Year를 나타내는 값은 'y'로 시작하고, 월Month을 의미하기 위해서는 'm'을 앞에 넣어주었으며, 날짜Day는 'd'를 사용하였다.('y2016': 2016년, 'm2': 2월, 'd20160205': 2016년2월5일) 그래서 파일이름도 HTML 태그에서 사용하는 id 값과 같게 하기 위해서 'd'를 날짜 앞에 추가하였다.

그림 4-11 블로그의 내용은 저장하는 파일은 날짜를 파일 이름으로 만들어서 관리한다.

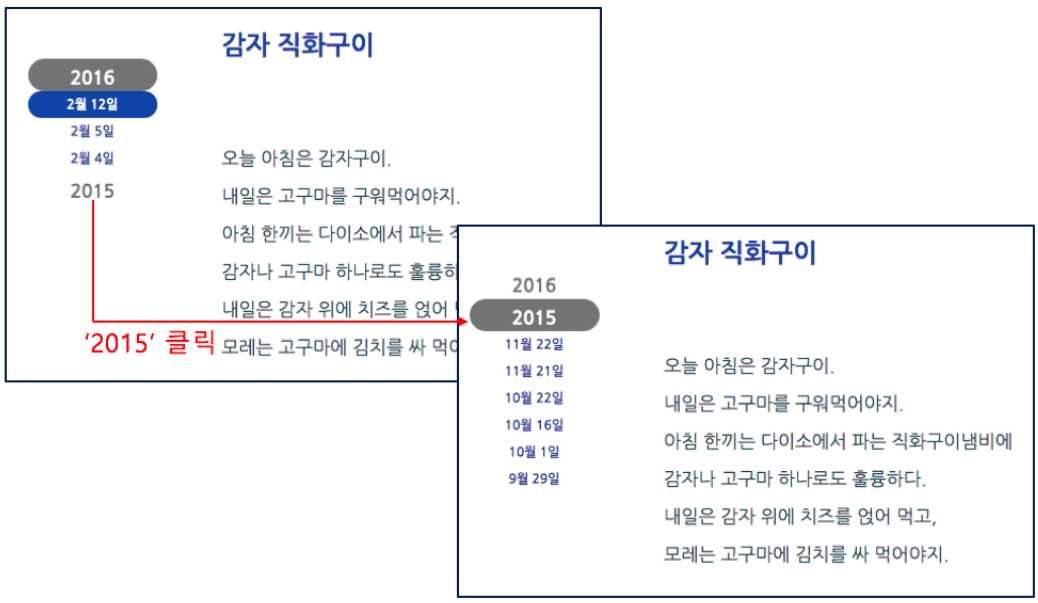


이전 장에서 만든 블로그 상단(Header 영역)에 있는 메뉴의 동작원리를 이해하였다면, 이번 장에서 만드는 Container 영역에 있는 메뉴의 동작원리도 동일함을 쉽게 이해할 수 있다.

[그림 4-12]는 연도 별로 보여지는 내용이 다른데, '2016'년의 내용을 보다가 '2015'를 클릭하면 기존의 항목들이 사라지고 '2015'년에 작성된 블로그 항목들이 보인다. 즉, 연도에 의존하는 태그들을 만들어서 CSS 디자인의 'display' 속성을 사용해서 보여주는 항목과 숨기는 항목들을 관리하면 된다. 웹 프로그래밍에서 태그 객체를 접근하기 위해서는 클래스나 id 값들을 사용하면 되기 때문에, 이번 장에서는 Container 영역에서 만들어지는 메뉴의 클래스와 id 값들의 연관성을 이해하면 된다.

[그림 4-12]에서 '2015'나 '2016'은 연도를 의미하는 객체이고 화면에서는 유일하게 존재하기 때문에 id를 사용해서 'y2015'와 'y2016'이라는 id 값으로 태그를 만들면 되며, 해당 연도에 종속되는 날짜 항목들은 다수가 존재하기 때문에 'y2015'와 'y2016'이라는 클래스 값으로 관리하면 된다. 즉, '2015'가 클릭되었다면 'y2016'라는 클래스 값을 가지는 항목들을 'display:none'을 사용해서 보이지 않게 하고, 'y2015'라는 클래스 값을 가지는 항목들을 'display:block'을 사용해서 브라우저에서 보이게 처리한다.

그림 4-12 블로그는 연도 별로 관리되며 연도를 클릭하면 새로운 항목들이 보여진다.



[코드 4-11]은 상단(Header 영역)의 하위 메뉴(ex: id="smenu_1_2")가 클릭되었을 때 서버로부터 가져오는 파일(ex: "smemu_1_2.html")의 코드를 보여주는데, 하위 메뉴의 이름을 파일이름으로 사용하고 있으며 이와 같은 방법으로 Container 영역의 메뉴들을 구성하면 된다. HTML 언어가 발전하면서 기존의 많은 태그들을 사용하지 않게 되었는데, 태그만을 사용해서 대부분의 웹 프로그래밍의 개발이 가능하기 때문이다. 실제 화면의 구성은 태그 위주로 만들 수 있으며, 간혹 텍스트의 디자인을 위해서 태그를 사용하면 된다. 코드에서 중요하게 봐야할 부분은 태그들의 클래스와 id 값들이 체계적으로 정리된 부분이다. 즉, 연도를 나타내는 태그의 id 값과 날짜를 나타내는 태그의 id 값은 코드에서 유일하게 한 번만 사용되며, 클래스 값으로는 중복되는 항목들을 사용하고 있다. 1.2장의 [그림 1-25]에서 설명한 클래스와 id의 개념을 이해하면 HTML 언어에서 태그들을 만들 때 많은 도움이 된다.

[코드 4-11]은 Container 영역에 보여지는 HTML 코드이고, [코드 4-12]는 화면에서의 이벤트에 대한 JavaScript 함수들을 보여준다. 날짜 별로 정리된 블로그 내용들은 메뉴에서 해당 날짜가 클릭되면 서버로부터 파일의 내용을 가져와서 [코드 4-11]의 아래에 있는 'container_second'라는 id 값을 가진 태그 객체에 넣어준다. 각각의 태그에 대한 CSS 디자인은 소스코드(/css/middlestory.css)에서 확인할 수 있기 때문에 여기서는 생략한다.

[코드 4-11] Container 영역에 보여지는 HTML 코드 (/container/smenu_1_2.html)

```
<script>
    clickedYear("y2016");                                          //---①
    clickedContainerMenu("d20160212", "2016");                     //---②
</script>

<div id="cmenu">
    <br>
    <br>
    <div class="year" id="y2016" onclick="clickedYear(this.id);">2016</div>      //---③
    <div class="y2016 m2 day" id="d20160212"                       //---④
      onmouseover="mouseOverContainerMenu(this.id);"
      onmouseout="mouseOutContainerMenu(this.id);"
      onclick="clickedContainerMenu(this.id, '2016');">2월 12일</div>
    <div class="y2016 m2 day" id="d20160205"
      onmouseover="mouseOverContainerMenu(this.id);"
      onmouseout="mouseOutContainerMenu(this.id);"
      onclick="clickedContainerMenu(this.id, '2016');">2월 5일</div>
    <div class="y2016 m2 day" id="d20160204"
      onmouseover="mouseOverContainerMenu(this.id);"
      onmouseout="mouseOutContainerMenu(this.id);"
      onclick="clickedContainerMenu(this.id, '2016');">2월 4일</div>
    <div class="year" id="y2015" onclick="clickedYear(this.id);">2015</div>
    <div class="y2015 m11 day" id="d20151122"
      onmouseover="mouseOverContainerMenu(this.id);"
      onmouseout="mouseOutContainerMenu(this.id);"
      onclick="clickedContainerMenu(this.id, '2015');">11월 22일</div>
    <div class="y2015 m11 day" id="d20151121"
      onmouseover="mouseOverContainerMenu(this.id);"
      onmouseout="mouseOutContainerMenu(this.id);"
      onclick="clickedContainerMenu(this.id, '2015');">11월 21일</div>
    <div class="y2015 m10 day" id="d20151022"
      onmouseover="mouseOverContainerMenu(this.id);"
      onmouseout="mouseOutContainerMenu(this.id);"
      onclick="clickedContainerMenu(this.id, '2015');">10월 22일</div>
    <div class="y2015 m10 day" id="d20151016"
      onmouseover="mouseOverContainerMenu(this.id);"
      onmouseout="mouseOutContainerMenu(this.id);"
      onclick="clickedContainerMenu(this.id, '2015');">10월 16일</div>
    <div class="y2015 m10 day" id="d20151001"
      onmouseover="mouseOverContainerMenu(this.id);"
      onmouseout="mouseOutContainerMenu(this.id);"
      onclick="clickedContainerMenu(this.id, '2015');">10월 1일</div>
    <div class="y2015 m9 day" id="d20150929"
```

```
        onmouseover="mouseOverContainerMenu(this.id);"
        onmouseout="mouseOutContainerMenu(this.id);"
        onclick="clickedContainerMenu(this.id, '2015');">9월 29일</div>
</div>
<div id="container_second"></div>                                    //---⑤
```

① 파일이 브라우저에서 열리면 자동으로 실행되는 JavaScript 코드를 <script></script> 태그 안에 넣으면 된다. 브라우저에서 파일을 다운받은 이후에 '2016'년의 버튼이 클릭된 것으로 처리하기 위해서 clickedYear("y2016") 함수를 실행했으며, 파일이 열릴 때 자동으로 보이는 항목을 이와 같이 조정할 수 있다.

② 자동으로 보이는 블로그의 내용을 'd20160212'로 지정하였으며, Container 영역의 메뉴 항목이 클릭될 때 실행되는 clickedContainerMenu() 함수는 날짜와 연도를 매개변수로 갖는다.

③ 태그의 디자인을 위해서 'year'라는 클래스 값을 가지며, 연도를 나타내는 태그의 고유한 id 값을 'y2016'과 같이 정하였다. Id 값은 브라우저에서 하나의 태그 객체에서만 사용된다는 것을 이해하는 것이 중요하다.

④ 메뉴에서 날짜 항목의 디자인을 위해서 'day'라는 클래스 값이 사용되었으며, 연도Year를 의미하는 'y2016'과 월Month을 의미하는 'm2'는 이벤트가 발생했을 때 객체의 접근을 위해서 주로 사용된다. 실제, 책에서 만든 블로그는 연도로만 파일을 구분하였는데, 블로그의 내용들이 많다면 월 단위로도 구분해서 파일을 관리하는 것이 좋다.

⑤ 날짜 별로 구분된 내용을 서버로부터 받아서 'container_second'라는 id 값을 가진 태그에 넣어준다. 태그의 크기와 디자인은 'middlestory.css' 파일에 정의되었다.

[코드 4-12] Container 영역 안의 메뉴에 대한 이벤트를 처리하는 JavaScript 코드 (/js/middlestory.js)

```
var currContainerMain;                                               //---①

function mouseOverContainerMenu(cmId) {                              //---②
    $("#"+cmId).css({ "color": "#ffffff", "background-color": "#1646a7" });
}

function mouseOutContainerMenu(cmId) {                               //---③
    if (cmId != currContainerMain) {
        $("#"+cmId).css({ "color": "##1646a7", "background-color": "#ffffff" });
    }
}
```

```
function clickedContainerMenu(cmId, year) {                                    //---④
    $("#"+currContainerMain).css({ "color": "#1646a7", "background-color": "#ffffff" });
    $("#"+cmId).css({ "color": "#ffffff", "background-color": "#1646a7" });
    currContainerMain = cmId;

    var strData;                                                               //---⑤
    if (year == null) {
        strData = currPage + "/" + cmId;
    } else {
        strData = currPage + "/" + year + "/" + cmId;
    }
    var param = { dataId: strData };
    var result = doAjax('./data/ajax_get_data.php', param);
    $("#container_second").empty().html(result);
}

var arrYear = [ 'y2015', 'y2016' ];                                            //---⑥

function clickedYear(year) {                                                   //---⑦

    for (var i=0; i<arrYear.length; i++) {
        if (year == arrYear[i]) {
            $('.'+arrYear[i]).css({ "display":"block" });
            $('#'+arrYear[i]).css({ "color": "#ffffff", "background-color": "#777777" });
        } else {
            $('.'+arrYear[i]).css({ "display":"none" });
            $('#'+arrYear[i]).css({ "color": "#777777", "background-color": "#ffffff" });
        }
    }
}
```

① 마우스가 Container 영역의 메뉴를 클릭했을 때, 현재Current 클릭한 메뉴의 id 값을 알기 위해서 currContainerMain 변수가 사용된다. 날짜 항목의 메뉴에서 이벤트가 발생했을 때 디자인 처리를 위해서 해당 메뉴의 id 값을 저장해야 한다.

② 마우스가 Container 영역의 메뉴 위로 이동할 때 실행되는 함수이며, 해당 메뉴를 활성화된 메뉴로 글자색과 바탕색을 변경한다.

③ 마우스가 메뉴 밖으로 이동할 때 실행되는 함수이며, 해당 메뉴가 클릭된 메뉴가 아니면 비활성화된 메뉴로 글자색과 바탕색을 변경한다.

④ 메뉴가 클릭되었을 때 실행되는 함수로서 활성화된 메뉴를 변경하고 서버로부터 해당 메뉴에 해당하는 파일의 내용을 받아와서 'container_second'라는 id 값을 가진 태그 안에 넣는다.

⑤ 서버로부터 읽어오기 위한 파일의 이름을 만드는데, 함수의 두 번째 매개변수인

year가 값을 가지고 있으면 현재 페이지(ex: currPage = "smenu_1_2")와 연도(ex: year = "2016")의 값을 포함하는 디렉토리 경로를 만들어서 서버로 보내게 된다. (ex: strData = "smenu_1_2/2016/d20160205") 이와 같이, 블로그의 내용을 서버로부터 요청할 때, 읽어야 하는 파일의 위치를 서버에게 알려준다.

⑥ 연도Year를 나타내는 태그를 처리하기 위해서 연도에 해당하는 값들을 arrYear 배열에서 관리하게 된다. 현재는 '2015'와 '2016'년 두 개의 값만을 갖고 있으나, 블로그가 오랫동안 유지될수록 이 값들은 계속 추가된다.

⑦ 연도가 클릭되었을 때 실행되는 함수로서 해당연도의 태그와 해당연도를 클래스 값으로 가지고 있는 종속된 태그들을 활성화시키고, 나머지는 비활성화를 시키고 종속된 날짜들은 화면에서 보이지 않게 한다.

블로그의 틀을 만들었으면 이후에 추가되는 내용들을 관리하는 것은 간단하다. 'Container' 영역의 메뉴를 'Copy & Paste'로 추가한 후에 변경해야 하는 id 값 등을 조정한 후에 서버에 블로그 내용을 가지고 있는 파일을 만들어주면 된다. [코드 4-13]과 [코드 4-14]는 블로그의 내용을 보여주는데, 모든 디자인은 '/css/middlestory.css' 파일에서 이미 정의되었기 때문에 간단한 HTML 코드만을 사용해서 날짜에 해당하는 파일에 내용을 만들면 된다. 예제에서는 이미지가 포함되기 때문에 조금 복잡하게 보이지만, 실제 대부분의 블로그는 텍스트 위주로 작성되기 때문에 윈도우의 워드Word 프로그램을 사용해서 내용을 작성한 이후에 서버 파일에 내용을 붙여넣고
 태그를 쉽게 추가해서 줄바꿈을 해주는 간단한 작업을 진행하면 된다. [코드 4-13]과 [코드 4-14]의 예제를 보여주는 이유는 [그림 4-13]과 [그림 4-14]에서와 같이 이미지의 테두리가 동그랗게 처리되는 방법을 보여주기 위해서다. 대부분의 사이트에서 제공하는 블로그 환경은 아직까지는 이미지를 화면에 넣고 디자인을 변경할 수 없지만, 직접 만들어서 관리하는 웹 블로그는 화면의 모든 디자인을 자신이 원하는 대로
변경하기 쉽다는 장점이 있다. 깔끔한 디자인을 위해서 자신에게 필요한 디자인 속성을 알고 사용하면 예쁜 웹 블로그를 꾸며 나갈수 있다.

[코드 4-13] 날자별로 관리되는 블로그 내용 (/container/smenu_2_1/d20160205.html)

```
<br><br><br>
<div class="big bold color1">거실에 한 줄기 빛</div>                    //---①
<br><br>
```

```
어둠이라는 좋은 배경을 활용할 줄 아는 조명.<br>
작년 여름 저녁, 옥상에서 고기 구워 먹을 때 사용했던 랜턴을<br>
요즘은 불 꺼진 거실에 덩그러니 켜둔다. <br>
<br>
<div style="position:relative;width:240px;height:360px;float:left;">                    //---②
    <img src="./img/smenu_1_2/2016/2/20160205_1.jpg" style="border-radius:50px;">
</div>
<div style="position:relative;left:30px;width:240px;height:360px;float:left;">
    <img src="./img/smenu_1_2/2016/2/20160205_2.jpg" style="border-radius:50px;">
</div>
<div style="clear:left;"></div>                                                    //---③
<br>
2016.2.5.<br>
```

① 공통적으로 사용되는 제목을 클래스 값을 사용해서 디자인 할 수 있다. 'big'은 큰 글자를 만들고, 'bold'는 굵은 글자로 만들고, 'color1'은 CSS 파일에서 정의한 값으로 글자색을 변경한다.

② <img> 태그 안에서 'border-radius' 속성을 50px로 설정해서 [그림 4-13]과 같이 모서리가 둥근 이미지로 만들었다. 이와 같이 픽셀 값을 정해서 둥근 모서리의 크기를 특정 값으로 설정할 수 있다.

③ 이미지의 태그들은 'float:left'로 설정되었기 때문에 'left' 설정 값을 해제하기 위해서 이와 같이 'clear:left' 속성만을 설정한 빈 태그를 넣었다.

그림 4-13 'border-radius' 디자인 속성을 사용해서 이미지의 테두리를 부드럽게 만들었다.

[코드 4-14] 날짜별로 관리되는 블로그 내용 (/container/smenu_2_1/d20160212.html)

```
<br><br><br>
<div class="big bold color1">감자 직화구이</div>
<br><br>
오늘 아침은 감자구이.<br>
내일은 고구마를 구워먹어야지.<br>
아침 한 끼는 다이소에서 파는 직화구이냄비에<br>
감자나 고구마 하나로도 훌륭하다.<br>
내일은 감자 위에 치즈를 얹어 먹고,<br>
모레는 고구마에 김치를 싸 먹어야지. <br><br>
<div style="position:relative;width:300px;">
    <img src="./img/smenu_1_2/2016/2/20160212_1.jpg" style="border-radius:50%;"> //---①
</div>
<br>
2016.2.12.<br>
```

① 이미지의 동그란 형태를 비율을 사용해서 변경할 수 있는데, <img> 태그 안에서 'border-radius' 속성을 50%로 설정하면 [그림 4-14]와 같이 이미지 전체가 동그랗게 처리된다.

그림 4-14 'border-radius' 디자인 속성에서 % 값을 사용해서 이미지를 동그랗게 만들었다.

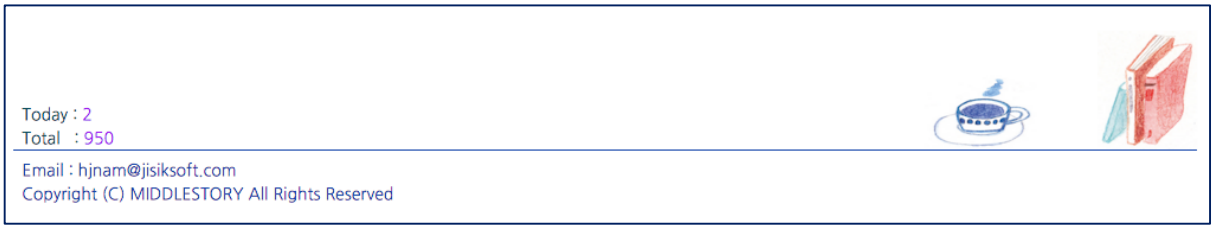
데이터베이스를 사용하지 않고 파일로만 만드는 웹 블로그는 웹 프로그래밍을 조금만 이해해도 쉽게 만들 수 있지만, 실제 웹 프로그래밍을 할 때는 많은 것을 알고 만들어야 하는 분야이기 때문에 여러 사람들이 모여서 분업화가 많이 이루어진다. 그러나, 전반적인 웹 프로그래밍의 기본 원리를 이해하면, 디자인만 변경해도 새로운 사이트를 만든 효과를 얻을 수 있으므로 이 책의 내용을 충분히 이해하고 자신만의 독창적인 웹 블로그를 독자들이 만들기를 바란다.

4.4 방문자수 카운트

블로그를 운영하면서 재미를 느끼는 부분은 방문자가 몇 명이나 되는지를 확인하는 것이다. 방문자가 많다면 블로거는 더 많은 재미를 느끼면서 내용을 자주 업데이트 하며 살아있는 블로그를 운영할 수 있다. [그림 4-15]는 웹 블로그의 화면 아래에 방문자 수를 보여주는 그림인데, 'Today'는 오늘 방문한 사람의 수를 보여주고 'Total'은 현재까지 방문한 전체 방문자수를 보여준다. 이번 장에서는 이와 같이 방문자 수를 저장하고 화면에 보여주는 방법을 설명한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_4/4.4/www/

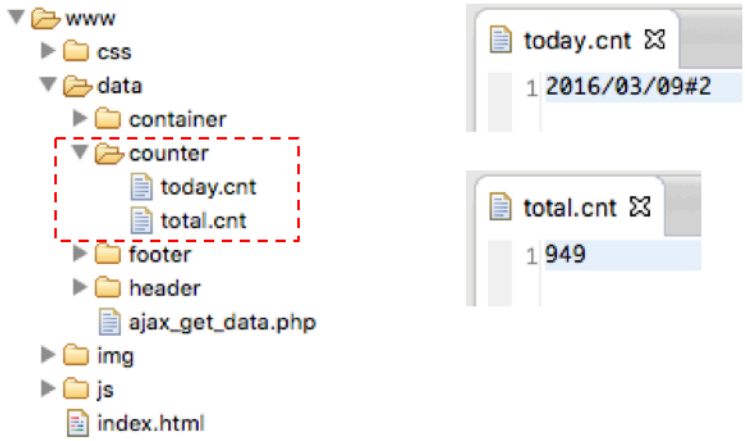
그림 4-15 Footer 영역의 왼쪽에 금일 방문자와 전체 방문자 숫자를 출력한다.



이 책에서 설명하는 웹 블로그는 데이터베이스를 사용하지 않고 파일로 모든 것을 관리하기 때문에 방문자 수를 저장하는 것도 파일을 만들어서 사용한다. [그림 4-16]은 방문자 수를 저장하고 있는 'today.cnt'와 'total.cnt'라는 두 개의 파일을

보여주는데, 'today.cnt' 파일에는 오늘 방문한 수를 저장하고 'total.cnt' 파일에는 블로그가 만들어진 이후에 방문한 전체 수를 저장한다. 'today.cnt' 파일에서 중요한 것은 현재 날짜도 저장하고 있어야 하는데, 날짜가 변경되면 해당 날짜를 바꿔주고 방문자수를 1로 초기화해야 한다. 문자열을 다루는 방법은 이전 Chapter의 '설문조사 만들기'에서 다루었는데, 여기서도 'today.cnt' 파일의 데이터는 구분자로서 '#' 문자를 사용해서 날짜와 방문자 수를 저장한다.

그림 4-16 방문자수가 저장되어 있는 두 개의 파일 내용



[코드 4-15]는 'Footer' 영역의 데이터를 브라우저에서 요청하면 'footer.html' 파일을 읽은 후, 데이터 앞에 방무자 수를 추가하는 코드를 넣어서 브라우저로 전송하는 PHP 코드를 보여준다. 블로그를 방문하는 사람의 브라우저에서는 'Header'와 'Footer' 영역의 데이터를 한 번만 가져가기 때문에, 방문자 수도 한 번만 증가하게 된다. 즉, 방문자가 블로그에 머물면서 다수의 내용을 본다 하더라도 방문자 수는 한 번만 카운트된다.

[코드 4-15] Footer 영역에 방문자수를 넣는 PHP 코드 (/data/ajax_get_data.php)

```
<?php

$dataId          = (isset($_POST['dataId'])) ? $_POST['dataId'] : "";

if ($dataId == "header")
        $filename = "./header/".$dataId.".html";
else if ($dataId == "footer")
        $filename = "./footer/".$dataId.".html";
else
        $filename = "./container/".$dataId.".html";

$fp = fopen($filename, "r");
```

```
$data = fread($fp, filesize($filename));
fclose($fp);

if ($dataId == "footer") {                                          //---①
    $fileCNT = "./counter/total.cnt";                               //---②
    $fp = fopen($fileCNT, "r");
    $cntTotal = fread($fp, filesize($fileCNT));
    fclose($fp);
    $cntTotal += 1;                                                 //---③
    $fp = fopen($fileCNT, "w");
    fwrite($fp, $cntTotal);
    fclose($fp);

    $today = date("Y/m/d");                                         //---④

    $fileCNT = "./counter/today.cnt";                               //---⑤
    $fp = fopen($fileCNT, "r");
    $dataToday = fread($fp, filesize($fileCNT));
    fclose($fp);
    $arrToday = explode("#", $dataToday);                           //---⑥
    if ($arrToday[0] == $today)                                     //---⑦
       $arrToday[1] += 1;
    else
       $arrToday[1] = 1;
    $dataToday = $today."#".$arrToday[1];                           //---⑧
    $fp = fopen($fileCNT, "w");
    fwrite($fp, $dataToday);
    fclose($fp);

    $data = '<div class="color2" style="position:relative;top:60px;width:100px;float:left;">
                  Today : <span class="color5">'.$arrToday[1].'</span></br>
                  Total   : <span class="color5">'
                .$cntTotal.'</span></div>'
                .$data;                                             //---⑨
}

print $data;
exit();

?>
```

① 'Footer' 영역의 데이터를 보내기 전에 방문자 수를 추가하며, 방문자 수가 저장된 파일에서 방문자 수를 읽고 1을 추가한 후 파일에 다시 저장한다.

② 전체 방문자 수를 읽기 위해서 'total.cnt' 파일을 읽어온다.

③ 전체 방문자 수를 1 증가시키고 파일에 다시 저장한다.
④ 오늘 날짜를 알기 위해서 PHP의 date() 함수를 사용하고 결과를 $today 변수에 저장한다.
⑤ 날짜와 하루동안의 방문자 수를 알기 위해서 'today.cnt' 파일의 데이터를 읽어온다.
⑥ '#' 구분자를 사용해서 날짜와 금일 방문자 수를 $arrToday 배열에 저장한다.
⑦ 파일에 저장된 날짜와 오늘 날짜가 같으면 방문자 수를 1 증가시키고, 날짜가 다르면 방문자 수를 1로 초기화한다.
⑧ 날짜와 변경된 방문자 수를 '#' 구분자를 기준으로 문자열로 만들어서 'today.cnt' 파일에 저장한다.
⑨ 'Footer' 영역의 내용 앞에 방문자 수를 보여주는 HTML 코드를 추가한 후, 브라우저로 보낸다.

웹 블로그는 회원을 받기 위한 '로그인'과 같은 페이지가 필요하지 않기 때문에 데이터베이스를 사용하지 않고 파일만을 사용해서 간단히 만들 수 있다. 웹 프로그래밍은 다른 프로그래밍에 비해서 개발하는 시간이 많이 소요되지만, 브라우저에서 즉시 내용을 확인할 수 있기 때문에 재미있게 프로그래밍하기 좋다. 여기서 만든 웹 블로그는 파일에 기반하기 때문에 Linux 운영체제에서 파일 접근에 대한 권한을 잘 설정해주어야 하는데, 파일을 읽고 쓰는데 문제가 있다면 3.2장에서 설명한 파일 접근권한을 다시 한번 읽기를 권한다.

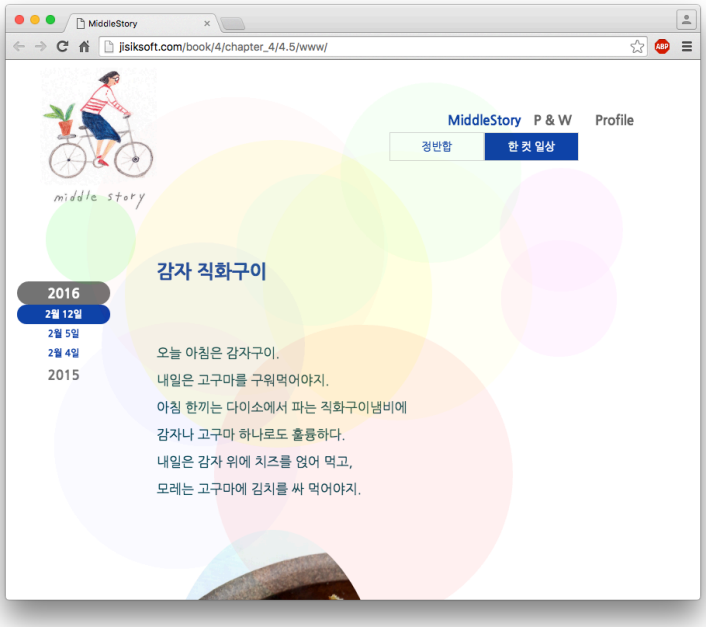
4.5 버블 효과 추가

이번 장에서는 블로그 화면에 효과를 주기 위해서 움직이는 버블Bubble을 구현하는데, [그림 4-17]과 같이 다양한 색의 12개 버블들이 화면에서 움직인다. 이 기능을 구현하기 위해서 1.4장의 [그림 1-65]에서 구현한 jQuery의 animate() 함수를 사용한 코드를 대부분 재사용했다. 이와 같이 브라우저에서 특정 효과를 나타내기 위해서는 JavaScript 언어로 추가 함수를 만들어서 구현한다. 화면에 효과를 주는 동작은 JavaScript 언어에서 12개 버블들의 디자인을 조금씩 변경하는 작업을 끊임없이

수행하기 때문에 CPU 사용률은 다소 높아진다.

확인 사이트: http://jisiksoft.com/book/4/chapter_4/4.5/www/

그림 4-17 화면에 버블이 움직이는 효과를 추가한 결과



[코드 4-16]은 12개의 동그라미들로 버블효과를 만들기 위해서 'bubble'이라는 id 값을 가진 태그를 만들었으며, makeBubble() 함수를 실행해서 'bubble'이라는 id 값을 가진 태그 안에 12개의 태그들을 생성하고 움직이게 만든다.

[코드 4-16] 버블을 만드는 태그를 추가하고 JavaScript 함수를 실행한다. (index.html)

```
<!DOCTYPE html>

<head>
    <title>MiddleStory</title>
    <meta charset="utf-8"></meta>
     <meta name="keywords" content="미들스토리, middlestory, middle, story"></meta>
    <link rel="stylesheet" href="./css/jquery-ui.min.css"></link>
    <link rel="stylesheet" href="./css/middlestory.css"></link>
    <link rel="stylesheet" href="./css/move_bubble.css"></link>                    //---①
    <script src="./js/jquery-2.1.3.min.js"></script>
    <script src="./js/jquery-ui.min.js"></script>
    <script src="./js/middlestory.js"></script>
```

```
    <script src="./js/move_bubble.js"></script>                    //---②
</head>
<body>
    <div id="header"></div>
     <div id="container"></div>
    <div id="footer"></div>

    <div id="bubble"></div>                                        //---③

    <script>
     $(window).bind("mousedown contextmenu", function() {
        return false;
     });

     initialize();
     makeBubble('bubble');                                         //---④
    </script>
</body>
</html>
```

① 움직이는 버블의 태그들의 디자인 속성을 가진 'move_bubble.css' 파일을 연결한다.

② 움직이는 버블의 태그들을 생성하고 움직이게 만드는 JavaScript 함수들을 가진 'move_bubble.js' 파일을 가져온다.

③ 움직이는 버블의 태그들을 포함하기 위한 'bubble'이라는 id 값을 가진 코드를 추가했다.

④ 움직이는 버블의 태그들을 생성하고 움직이게 만드는 makeBubble() 함수를 실행한다.

[코드 4-17]은 움직이는 버블의 태그들을 생성하고 jQuery의 animate() 함수를 실행해서 <dIv> 태그들의 위치와 크기및 투명도를 변경해서 움직이는 효과를 만드는 JavaScript 코드를 보여준다. 1.4장의 [코드 1-39]에서 구현한 코드와 구현 방법은 거의 일치하며, 브라우저의 크기가 변경될 때마다 발생하는 'resize' 이벤트에서 버블이 움직이는 영역을 변경하고 버블이 생성된 이후에 위치를 초기화하기 위해서 initLocation() 함수를 추가했다.

[코드 4-17] 태그를 생성하고 animate() 함수로 디자인을 변경한다. (/js/move_bubble.js)

```
var width, height;
var numCircle = 12;                                                //---①
```

```
functin move(container) {                                          //---②

    var opacColor = (Math.random() * 0.09) + 0.01;                 //---③
    var posX = Math.random() * width;                              //---④
    var posY = Math.random() * height;
    var moveTime = Math.random() * 10000 + 7000;                   //---⑤
    var sizeCircle = (Math.random() * 400) + 50;                   //---⑥

    var distWidth = 0;                                             //---⑦
    if ((posX + sizeCircle) > width)
        distWidth = width - sizeCircle;
    var distHeight = 0;
    if ((posY + sizeCircle) > height)
        distHeight = height - sizeCircle;

    $( "#"+container ).animate({                                   //---⑧
                                opacity: opacColor,
                                left: (posX - distWidth)+'px',
                                top: (posY - distHeight)+'px',
                                width: sizeCircle+'px',
                                height: sizeCircle+'px'
                                }, moveTime);

    setTimeout(function(){ move(container); }, moveTime);          //---⑨
}

function initLocation(container) {                                 //---⑩

    var opacColor = (Math.random() * 0.09) + 0.01;
    var posX = Math.random() * width;
    var posY = Math.random() * height;
    var sizeCircle = (Math.random() * 300) + 100;

    var distWidth = 0;
    if ((posX + sizeCircle) > width)
        distWidth = width - sizeCircle;
    var distHeight = 0;
    if ((posY + sizeCircle) > height)
        distHeight = height - sizeCircle;

    $( "#"+container ).css({
                                opacity: opacColor,
                                left: (posX - distWidth)+'px',
                                top: (posY - distHeight)+'px',
                                width: sizeCircle+'px',
```

```
                height: sizeCircle+'px'
            });
}

function makeBubble(container) {                                    //---⑪

    width = $(window).width();
    height = $(window).height();

    var i;
    var str = '';
    for (i=0; i<numCircle; i++) {                                   //---⑫
     str += '<div class="circle" id="circle_'+i+'"></div>';
    }
    $('#'+container).html( str );

    for (i=0; i<numCircle; i++) {                                   //---⑬
     initLocation('circle_'+i);
    }
    for (i=0; i<numCircle; i++) {                                   //---⑭
        move('circle_'+i);
    }
}

$(window).bind("resize", function() {                               //---⑮
    width = $(window).width();
    height = $(window).height();
});
```

---

① 화면에 만들어지는 버블의 갯수를 12로 정했다.

② 버블을 움직이게 만드는 함수로서 매개변수 'container'는 하나의 버블을 만들기 위해 생성된 태그의 id 값을 갖는다.

③ 버블의 투명도Opacity를 만드는 함수로서 Math.random() 함수는 [0, 1] 사이의 실수를 만드는데, 이 값에 0.09를 곱하고 0.01을 더하기 때문에 opacColor 변수는 [0.01, 0.1] 사이의 값을 갖는다. 투명도가 1이면 투명도가 없는 것이고, 값이 작아질수록 투명도가 높아진다.

④ 버블의 (x, y) 위치값을 Math.random() 함수를 사용해서 만들고 posX와 posY 변수에 저장한다.

⑤ 버블의 위치가 변하는 시간을 계산해서 moveTime 변수에 저장하는데, 7초와 17초 사이의 값으로 만들기 위해서 Math.random() 함수에 10000을 곱하고 7000을 더했다.

⑥ 버블의 크기가 변경되는데, 가장 작은 것은 50px이고 가장 큰 버블은 450px의 지름을 가지게 하기 위해서 Math.random() 함수에 400을 곱하고 50을 더했다.

⑦ 원이 브라우저의 오른쪽 밖으로 나가는 경우에는 distWidth 변수가 0보다 큰 값을 갖고, 아래쪽 밖으로 나가는 경우에는 distHeight 변수가 0보다 큰 값을 갖는다. 이후에 해당 변수의 값이 적용되어 버블의 위치가 결정된다.

⑧ $("#"+container) 함수는 해당 버블의 태그 객체를 가져오고, animate() 함수를 사용해서 moveTime 변수에 저장된 시간동안 객체의 디자인을 새로운 값으로 서서히 변경한다.

⑨ animate() 함수에서 디자인이 변경되는 시간인 moveTime 변수의 값을 가지고 setTimeout() 함수를 실행한다. moveTime 변수의 값(milli-second)이후에 move() 함수를 실행하기 때문에, 버블은 animate() 함수의 종료 이후에 곧바로 move() 함수가 다시 실행되는 효과를 만들었다.

⑩ 버블이 처음 생성되고 위치와 크기를 정하는 함수로서, 위에서 구현한 move() 함수와 거의 같은 코드내용을 가지고 있는데 move() 함수에서는 animate() 함수를 사용해서 디자인을 서서히 변경하지만 여기서는 css() 함수를 사용해서 디자인을 즉시 변경한다.

⑪ 버블을 만들기 위해서 태그들을 생성하고 움직이게 만드는 함수로서, 'index.html' 파일에서는 이 함수만을 실행해서 움직이는 버블을 만들었다.

⑫ 12개 버블들을 만들기 위해서 for loop을 12번 실행하고, 'circle'이라는 클래스 값과 고유한 id 값을 가진 12개의 태그들을 만들어서 'container'(여기서는 'bubble'이라는 값을 가지고 있다.) id 값을 가지고 있는 태그 안에 넣는다.

⑬ 생성된 12개 태그들의 위치와 크기를 초기화한다.

⑭ 생성된 12개 태그들을 서서히 움직이게 하기 위해서 move() 함수를 실행한다.

⑮ 브라우저의 크기가 변경될 때마다 발생하는 'resize' 이벤트는 변경된 브라우저의 가로와 세로 길이를 width와 height 변수에 저장한다. 움직이는 버블은 항상 width와 heigh 변수를 가지고 위치가 계산되기 때문에 브라우저 전체에서 움직이는 효과를 가진다.

[코드 4-18]은 생성된 버블 태그들의 CSS 디자인 속성을 보여주는데, 원을 만들기

위해서 'border-radius' 속성을 '50%'로 설정했으며 버블이 마우스로 클릭하는 메뉴부분을 가렸을 때에도 메뉴를 클릭할 수 있게 'pointer-events' 속성을 'none'으로 설정했다.

[코드 4-18] 생성된 버블 태그들의 디자인 속성값을 설정한다. (/css/move_bubble.css)

```
.circle {
    position:fixed;                                    //---①
    border-radius:50%;                                 //---②
    pointer-events:none;                               //---③
}
#circle_1 {                                            //---④
    background-color:#ff0000;
}
#circle_2 {
    background-color:#00ff00;
}
#circle_3 {
    background-color:#0000ff;
}

##### 생략 #####
```

① 모든 버블의 'position' 속성은 'fixed' 값을 갖기 때문에 화면이 위아래로 이동해도 버블들은 브라우저 안에서만 움직인다. 웹 프로그램을 만들면서 'fixed' 값은 사용될 일이 별로 없는데, 움직이는 버블을 만들 때는 꼭 필요한 설정값이다.

② 태그는 사각형 영역을 가지는데, 'border-radius' 속성을 사용해서 원형을 만들 수 있으며 여기서는 '50%' 설정 값을 사용해서 원을 만들었다.

③ 움직이는 버블에서 꼭 필요한 설정 값으로, 'pointer-events' 속성을 'none'으로 설정해서 버블이 메뉴항목 위에 있을때 마우스로 메뉴들을 클릭할 수 있다.

④ 모든 버블을 만드는 태그들은 고유한 id 값을 가지고 있으며, id 값을 가지고 버블들의 색을 설정한다.

일반적인 블로그와 차별화를 두기 위해서 '움직이는 버블' 효과를 구현했다. '움직이는 버블' 효과를 사용할 필요가 없으면 이번 장의 내용을 추가하지 않으면 되고, 자신만의 특별한 효과를 주기 위해서는 JavaScript 언어로 새로운 함수를 만들면 된다. 브라우저에서 동작하는 작은 기능 하나를 만들 때에도 JavaScript 언어로 긴 코드를 구현해야 할 때가 많은데, 브라우저에서 표현하는 기능들은

시각화할 수 있기 때문에 서버에서만 동작하는 프로그램보다 재미있을 때가 많다. 브라우저에도 다양한 기능을 할 수 있게 웹 프로그래밍 언어는 지금도 발전하고 있으며, 발전할수록 좀더 전문적인 프로그래밍을 필요로 할수가 있다. 예를 들면, jQuery의 animate() 함수의 기능은 간단할 수 있지만, 이 기능을 사용해서 다양한 효과를 만드는 것은 프로그래머의 역할이다.

chapter 5

홈페이지 만들기

이번 Chapter에서는 회사 홈페이지를 만드는 방법을 설명한다. 데이터베이스Database와의 연동에 초점을 두었는데 데이터베이스 테이블을 만들고 데이터를 저장하고 가져오는 방법들을 이해해야 한다. 지금까지의 내용들을 충분히 이해한 독자라면 데이터베이스를 이해하고 난 후 대부분의 웹 사이트의 구조를 쉽게 이해할 수 있을 것이다. 웹 프로그래밍을 하는 많은 사람들이 어려워하는 부분이 데이터베이스지만, 데이터베이스까지 이해하면 대부분의 회사에서 웹 사이트를 운영하는데 큰 무리가 없다고 본다. 이 책에서는 PHP 언어를 가지고 MySQL이라는 데이터베이스를 사용하지만, JSP를 사용한다고 하더라도 PHP의 사용방법과 기본 개념은 같다. 즉, 서버에서 어떠한 프로그래밍 언어를 사용하더라도 데이터베이스에 연결Connect하고, SQL(Structured Query Language) 언어를 사용해서 원하는 데이터를 가져오거나 저장하면 된다. SQL 언어는 모든 데이터베이스에서 공통적으로 사용하는 데이터베이스 언어인데, 다양한 종류의 데이터베이스(ex: Oracle, MySQL, MSSQL)는 독자적으로 설계되었지만 데이터베이스에 접근해서 데이터를 조작하는 방법은 모든 데이터베이스가 하나의 SQL 언어만을 사용한다. MySQL은 무료인데 Oracle 회사에서 인수한 이후로 업데이트가 이루어지지 않아서, 많은 사용자가 MySQL의 다음 버젼이라고 할 수 있는 MariaDB 데이터베이스를 사용하는 추세이다. 하지만, MySQL은 잘 만들어진 데이터베이스이고 Linux 서버에서 아직도 많이 사용하고 있기 때문에, 이 책에서는 MySQL을 사용해서 데이터베이스를 구축하였다.

5.1장에서는 'Chapter 4'의 웹 블로그와 같은 방법으로 만들어진 '지식소프트' 회사의 홈페이지를 설명하고, 5.2장부터는 데이터베이스를 사용해서 홈페이지의 데이터를 저장하고 가져오는 방법을 구현하였다. 5.2장에서는 홈페이지의 방문자 수를 파일에 저장하지 않고, 데이터베이스에 테이블을 만들어서 관리하는 방법을 설명하였다. 5.3장에서는 사용자 정보를 데이터베이스에 저장하고 PHP를 사용해서 로그인하는 방법을 설명하는데, 로그인이 되었다는 것은 사용자 PC와 서버 간에 세션Session이 연결

되었다는 의미이다. 대부분의 홈페이지에서 로그인을 하면 이후부터는 서버에서 연결된 사용자가 누구인지를 알게 되며, 사용자는 정상적인 사용자로 인식되어 서버에 저장된 데이터들을 합법적으로 접근할 수 있게 된다. 예를 들면, 대학교에서 로그인을 한 후에 '수강신청'과 같은 서비스를 이용할 수 있게 되고 자신이 수강하는 교과과목을 정하고 검색할 수 있다. 또한 '온라인 마켓' 사이트에서도 물건을 주문하기 위해서는 정상적인 사용자로 로그인을 해야 한다. 5.4장에서는 '게시판'을 만드는 방법을 구현하였는데, 게시판의 내용은 파일로 관리할 수 없고 데이터베이스에 내용을 저장해야 한다. 브라우저에서 보여지는 게시판은 서버로부터 정보를 가져와서 화면에 테이블 형태로 보여주고, 클릭된 항목의 내용은 서버로부터 다시 가져와서 화면에 보여주는 것으로 이해하면 된다. 이 책에서는 게시판을 단순하게 만들었으며, 이 책을 이해한 독자라면 자신만의 새로운 디자인으로 게시판을 꾸밀 수 있을 것이다.

5.1 지식소프트 홈페이지

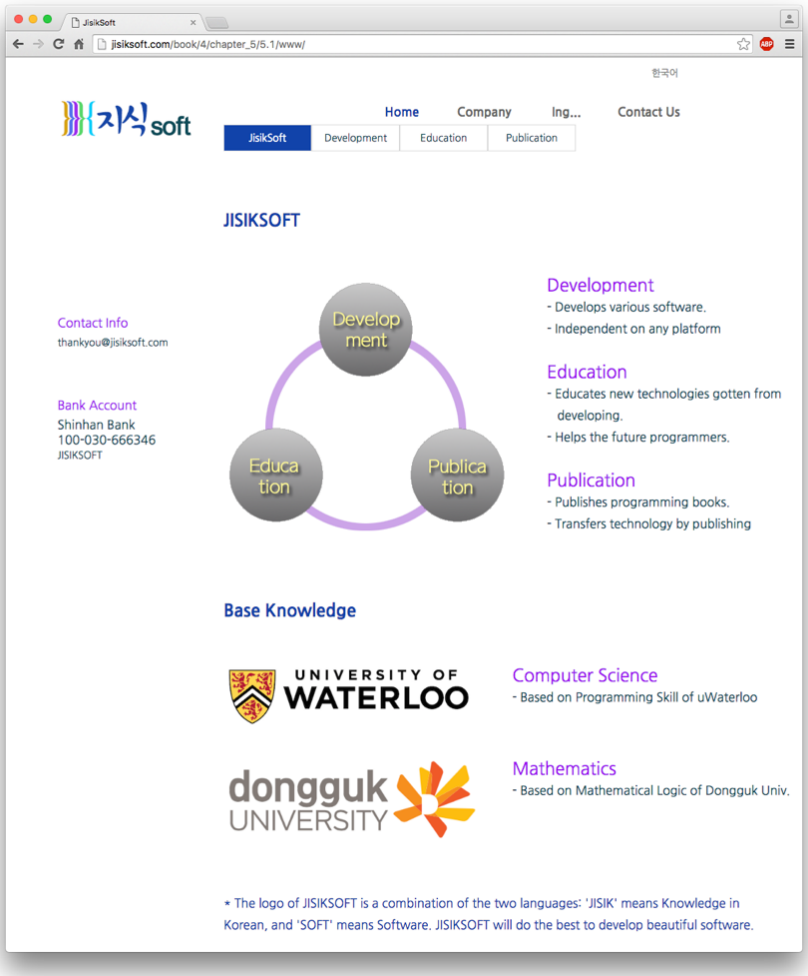
이 책에서는 웹 프로그래밍에서 기본적으로 필요한 항목들만을 가지고 브라우저에서 동작하는 웹 사이트나 프로그램을 만드는 방법을 설명한다. 오래전부터 발전되어 온 HTML 언어는 많은 종류의 태그들을 정의해 놓았는데, 웹 프로그래밍을 어느 정도 하는 사람들은 태그만 가지고도 얼마든지 웹 프로그래밍을 만들 수 있다. 이번 Chapter에서도 태그만으로 홈페이지를 만드는데 글자의 디자인을 위해서 가끔 태그를 사용한다. 이제부터 구현하는 작은 회사인 '지식소프트'의 홈페이지는 웹에서 서비스를 하고 있는 웹 사이트이며, 작은 홈페이지가 어떻게 만들어질 수 있는지를 독자는 알게 될 것이다. 물론, 이 책에서 웹 프로그래밍을 만드는 방법은, 오랫동안 웹 프로그래밍을 해온 필자가 독자 입장을 고려하여 최대한 이해하기 쉬운 방법으로 만든 것이며, '지식소프트'가 출간한 이전 책들에서 설명한 것과 같이 프로그래밍에는 정답이 없기 때문에 독자는 이 책의 방법을 이해한 후에 자신만의 방법으로 새로운 홈페이지를 만들었으면 한다.

[그림 5-1]은 이전 Chapter에서 만든 웹 블로그와 같은 방식으로 만든 '지식소프트' 홈페이지인데 데이터베이스를 사용하기 전 단계까지 만들어진 상태이다. 이번 장에서는 홈페이지의 대략을 만들고, 다음 장부터는 데이터베이스를 사용하는 방법을 설

명한다.

확인 사이트: http://jisiksoft.com/book/4/capter_5/5.1/www/

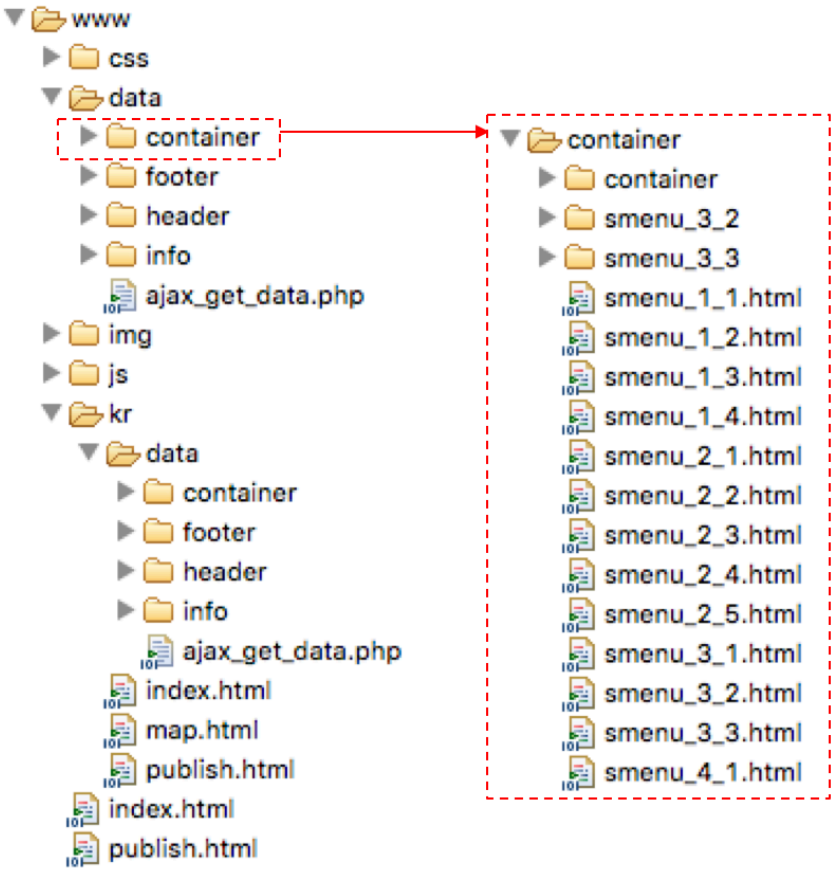
그림 5-1 이전 Chapter 의 웹 블로그와 같은 방식으로 만든 '지식소프트' 홈페이지



이전 Chapter의 웹 블로그와 마찬가지로 홈페이지의 대부분의 내용은 파일구조로 이루어져 있으며, 홈페이지의 가운데 부분에 들어가는 내용들은 '/data/container/' 디렉트리에 정리하였다. 파일의 이름은 홈페이지의 상단의 메뉴 태그의 id 값과 연관되었으며, [그림 5-2]는 홈페이지의 파일 구조를 보여준다. '지식소프트' 홈페이지는 영문과 한글 두 개의 언어로 서비스를 제공하는데, 영문 버전은 '/www/' 디렉토리에 모두 정리했고 한글 버전은 '/www/kr/' 디렉토리에 한글 내용의 파일을 보관하고 있다. 즉, 하나의 홈페이지에서 다양한 나라의 서비스를 제공하더라도 기본적으로 동작하는 CSS와 JavaScript 파일들은 공통적으로 사용하고, 나라마다 다르게 제공되는 번역된 내용만을 새로운 대렉토리에서 관리하면 된다. 물론, 규모가 큰 다국적 회사들은 각 나라마다 새로운 시스템 환경을 구축해서 관리하겠지만, 소규모의 회사라면

이와 같이 언어만을 다르게 해서 관리하는 것이 개발에 소요되는 비용과 시간을 줄일 수 있다. '지식소프트' 홈페이지는 필자 혼자서 만든 것이기 때문에 최대한 간단히 만들려고 노력한 것이며, 독자들은 웹 사이트를 이해하기 위한 하나의 방법으로 이해하기를 바란다.

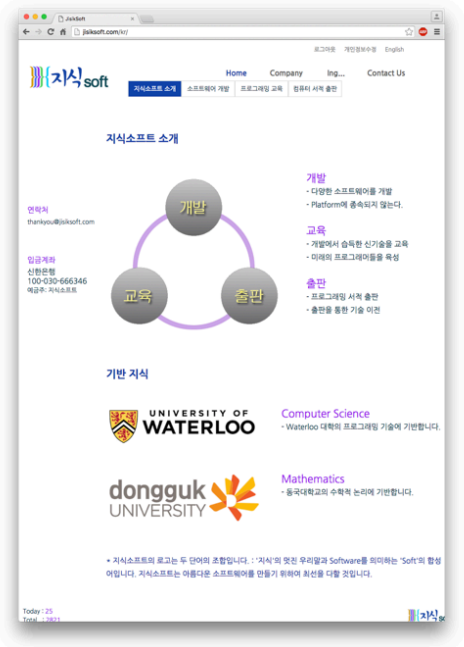
그림 5-2 홈페이지 시스템의 파일구조



'지식소프트' 홈페이지는 접속한 이후에 브라우저 상단의 주소 창에 있는 내용이 변하지 않는 구조인데, 홈페이지의 내용이 변경되어도 처음에 자동으로 실행되는 'index.html' 파일이 변경되지 않기 때문이다. 그러나, 많은 사이트에서는 처음 접속할 때 자동으로 보이는 페이지를 변경하는 것이 필요한데, [그림 5-3]은 'index.html' 파일 대신에 'publish.html' 파일을 추가해서 홈페이지의 첫 화면에서 특정 페이지가 열리는 것을 보여준다. 일반적으로 회사의 홍보를 위해서 만드는 페이지는 이와 같이 특정 파일을 하나 추가해서 만든 후에 파일 이름까지 브라우저의 주소 창에 넣어서 실행시키는 구조를 갖는다.

그림 5-3 특정 페이지를 첫 화면에서 보이도록 만들 수 있다.

http://jisiksoft.com/kr/



http://jisiksoft.com/kr/publish.html

[코드 5-1]은 이전 Chapter의 '웹 블로그 만들기'의 'index.html' 파일의 내용과 대부분이 동일하며, 마지막에 'Container 영역'에서 열리는 내용을 특정 페이지에서 가져오는 JavaScript 함수를 추가하였다. 코드의 상단에 추가된 js 파일은 두 가지인데, 'jquery.ui.touch-punch.min.js' 파일은 스마트폰이나 태블릿 PC에서의 터치를 위해서 추가한 것이며 'http://maps.googleapis.com/maps/api/js' 파일은 홈페이지에서 지도를 보여주기 위해서 Google의 지도 서비스를 추가했다.

[코드 5-1] '지식소프트'의 출간된 서적을 보여주는 페이지로 이동하는 HTML 코드 (/kr/publish.html)

```
<!DOCTYPE html>

<head>
    <title>JisikSoft</title>
    <meta charset="utf-8"></meta>
     <meta name="keywords" content="지식소프트, jisiksoft, IT, Programming"></meta>
     <meta name="Description" content="대한민국 IT 발전을 위하여 노력합니다."></meta>
    <link rel="stylesheet" href="../css/jquery-ui.min.css"></link>
    <link rel="stylesheet" href="../css/jisiksoft.css"></link>
    <script src="../js/jquery-2.1.3.min.js"></script>
    <script src="../js/jquery-ui.min.js"></script>
    <script src="../js/jquery.ui.touch-punch.min.js"></script>                    //---①
    <script src="http://maps.googleapis.com/maps/api/js"></script>                //---②
    <script src="../js/jisiksoft.js"></script>
```

```
</head>
<body>
    <div id="header"></div>
    <div id="info"></div>
     <div id="container"></div>
    <div id="footer"></div>

    <script>
        $(window).bind("mousedown contextmenu", function() {                    //---③
                return false;
        });

        initialize();                                                            //---④

        mouseOverMenu("menu_2");                                                 //---⑤
        clickedSubMenu("smenu_2_3");
    </script>
</body>
</html>
```

① 브라우저에서 동작하는 웹 프로그래밍은 개발할 때는 PC의 브라우저 환경을 사용하지만, 최근 사용자들은 스마트폰이나 테블릿 PC에서 인터넷에 접속할 때가 많다. 일반 PC 브라우저에서는, 마우스를 클릭하는 것으로 대부분의 이벤트를 발생하지만, 스마트폰이나 테블릿 PC에서는 화면의 터치Touch를 사용해서 이벤트를 발생한다. 'jquery.ui.touch-punch.min.js' 파일은 마우스가 아닌 화면의 터치를 사용해서 이벤트가 발생하는 기기에서 정상적으로 동작하게 하기 위해서 추가한 것이며, 모든 기기에서 동작하는 웹 프로그램을 만드는 프로그래머에게 상당히 편리한 JavaScript 라이브러리라고 할 수 있다.

② Daum과 Naver와 같은 포털 서비스는 웹 사이트에 지도를 넣을 수 있는 서비스를 제공하고 있다. 하지만, 국내에서 제공하는 서비스의 단점은 현재까지는 대한민국 지도만을 만들 수 있는 방법만을 지원한다. 하지만, Google에서는 세계지도를 만들수 있는 방법을 제공하는데, 해외에서 세계지도를 봐야하는 사이트를 개발하기 위해서는 Google의 지도 서비스를 이용해야 한다. 브라우저에서의 이벤트를 위한 프로그래밍은 JavaScript 언어만을 사용해야 하기 때문에, Google에서 제공하는 js 파일의 JavaScript 라이브러리를 사용해서 지도를 만든다고 이해하면 된다.

③ 마우스를 이용해서 내용을 복사하는 것을 금지시킨다. 마우스를 눌렀을 때(mousedown)와 마우스 오르쪽 버튼을 눌렀을 때 열리는 브라우저의 메뉴(contextmenu)에 대한 이벤트에서 아무런 동작도 하지 않게 처리하면 된다.

④ 'index.html' 파일에서와 같이 initialize() 함수는 페이지의 모든 내용을 가져온다.

즉, 여기까지의 내용은 'index.html' 파일의 내용과 다른 점은 없으며, 아래의 mouseOverMenu() 함수와 clickedSubMenu() 함수를 사용해서 특정 페이지를 다시 불러오는 동작을 추가한다.

⑤ mouseOverMenu("menu_2") 함수를 사용해서 메뉴의 두 번째 항목에 마우스가 옮겨지고, clickedSubMenu("smenu_2_3") 함수를 사용해서 하위 메뉴의 세 번째 항목이 클릭된 효과를 만들었다. 이렇게 해서 'Conatiner 영역'에 특정 파일의 내용을 보여줄 수가 있는데, 이렇게 하면 코드를 쉽게 이해할 수 있어 코드 관리가 용이하다.

확인 사이트: http://jisiksoft.com/book/4/chapter_5/5.1/www/kr/publish.html

그림 5-4 홈페이지의 특정 내용을 자동으로 보여준 결과

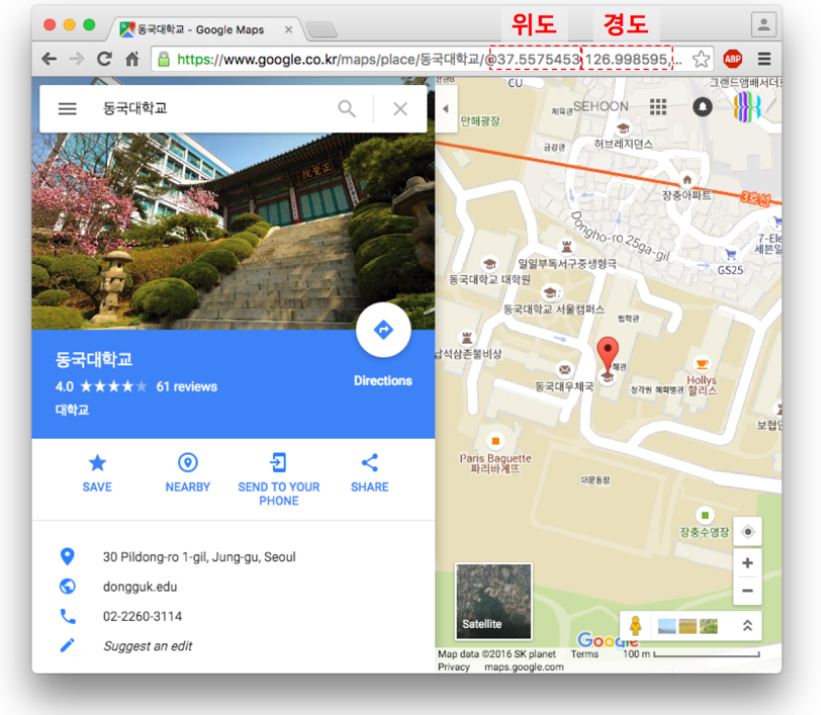
회사의 약도를 Google 지도 서비스를 이용하면 웹 페이지에 쉽게 넣을 수 있다. 구글은 웹 개발자들을 위하여 '구글 맵 API' 사용법을 설명한 페이지를 제공한다.(https://developers.google.com/maps/) 브라우저에서 동작하는 지도 서비스를 이용하기 위해서는 JavaScript로 만들어진 API를 사용해야 하는데, JavaScript API를 사

용한다는 것은 JavaScript 함수들로 이루어진 지도 라이브러리Library를 사용해서 구글의 지도 서비스를 사용한다고 이해하면 된다. [그림 5-5]는 Google의 지도 서비스를 사용하여 홈페이지에 약도를 넣은 결과를 보여주는데, 회사의 위치와 정보를 지도에서 보여주고 마우스의 휠을 사용해서 지도의 크기를 조절할 수 있다. 만약, 구축하는 웹 사이트에서 한국에 있는 사용자만을 위해서 위치 정보를 제공한다면 Daum과 Naver에서 제공하는 지도 서비스를 이용하는 것이 좋고, 외국에서 보여지는 사이트를 만든다면 Google의 지도 서비스를 사용하면 된다. 우리나라에서는 Google의 지도 서비스를 좋아하지 않는 편인데, Google Map에서는 '동해'를 '일본해'로 표시하기 때문이다. 이 책에서는 영문으로 만든 페이지가 있고 지도를 위한 API의 사용법을 이해하는 목적으로 Google 맵을 사용했음을 독자가 이해해 주었으면 한다. 다른 회사에서 제공하는 지도를 위한 API를 사용하더라도 기본은 지구에서의 위치 값으로 사용되는 '위도'(Latitude)와 '경도'(Longitude)를 기본으로 사용한다는 것을 이해하면 된다. 즉, 자신이 원하는 위치의 '위도'와 '경도'를 정확히 알고 화면의 가운데에 특정 위치가 중심이 되어서 지도가 그려지는 것이며, 지도의 크기와 지도에 출력하는 내용을 API의 함수를 사용해서 프로그래머가 정한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_5/5.1/www/kr/map.html

그림 5-5 Google 의 지도 서비스를 사용하며 홈페이지에 약도를 넣은 결과

그림 5-6 Google Map 에서 동국대학교를 검색한 결과 주소 창에 보여지는 위도와 경도



[코드 5-2]는 [그림 5-5]의 내용을 만드는 HTML 코드이며, 지도를 만들기 위해서 JavaScript 언어로 Google의 지도 API를 사용한다. 코드의 내용에서 '지식소프트'의 위도와 경도는 (37.5815956, 127.0129221) 값을 갖는데, 필자는 지도를 만들 때 원하는 위치의 정확한 값을 찾는 것도 재미있었다. 지도를 만들 때 대략적인 위치는 Google Map(http://map.google.com/)의 검색 창에서 주소를 검색하면 브라우저의 주소 창에서 해당위치의 위도와 경도 값을 확인할 수 있는데, [그림 5-6]은 Google Map에서 '동국대학교'를 검색한 결과를 보여준다. [코드 5-2]는 Google의 지도 API를 사용해서 위도와 경도를 넣고 지도의 크기를 확대(Zoom)하고 위성 사진이 아니라 보통의 지도(RoadMap) 사진을 보여주며 지도에서 표시하는 위치에 설명문구를 넣어주는 JavaScript 코드를 보여준다. 프로그래밍을 하다보면 많은 API들을 보게 되는데, 쉽게 이해하자면 사용하는 방법은 보통의 라이브러리와 같으며 API에 정의된 함수를 사용해서 API를 제공하는 회사의 기능을 사용한다고 생각하면 된다.

[코드 5-2] 구글맵 API 를 사용해서 지도 만들기 (/kr/data/container/smenu_4_1.html)

```
<br><br>
<div class="big color1">전국의 모든 Cafe는 지식소프트의 회의실입니다.</div>
<br>
<div class="big color5">02-743-0367</div>
<div class="big color5">thankyou@jisiksoft.com</div>
<br><br>
```

```
<div id="googleMap" style="width:700px;height:500px;"></div>
<div class="color5">서울 성북구 낙산길 243-15, 101-1004</div>
<br><br>

<script language=javascript>

    var myCenter=new google.maps.LatLng(37.5815956,127.0129221);          //---①

    var mapProp = {                                                       //---②
      center:myCenter,
      zoom:16,
      mapTypeId:google.maps.MapTypeId.ROADMAP
    };

   var map=new google.maps.Map(document.getElementById("googleMap"),mapProp);//---③

    var marker=new google.maps.Marker({                                   //---④
      position:myCenter,
    });

    marker.setMap(map);                                                   //---⑤

    var infowindow = new google.maps.InfoWindow({                         //---⑥
      content:"지식소프트는 한성대학교 옆에 있습니다."
    });

    infowindow.open(map,marker);                                          //---⑦

</script>
```

① 위도와 경도 값을 저장하고 있는 myCenter 클래스 변수를 만든다. Google 지도 API에서 만들어 놓은 함수를 사용해서 클래스로 만드는 것이며, 쉽게 이해하자면 "var myCenter = { lat: 37.5815956, lng: 127.0129221 };"와 같은 의미로서 JSON 데이터로 표현된 값을 갖게 된다.

② 지도의 속성Property을 정의하기 위해서 JSON 데이터 형식으로 지도의 가운데 위치 값, 지도를 확대한 값, 그리고 지도의 형식Type을 정의하였다.

③ 속성을 넣어서 'googleMap'이라는 id 값을 갖는 태그 객체에 지도를 만든다. 만들어진 지도는 새로운 객체가 되며, map 변수가 가리키게 된다. Google 지도 API의 Map() 함수는 클래스를 만들어주는 생성자이다.

④ 지도에 정보를 보여주기 위해 사용되는 클래스 함수는 Marker()이며, 'position'

속성을 사용해서 위치 값을 넣어준다.

⑤ Map() 클래스 객체인 map 변수를 매개변수로 사용해서 Marker() 클래스의 setMap() 함수를 사용해서 화면에 보여지는 지도 객체와 연결한다.

⑥ 지도의 화면Window 가운데에 위치에 대한 정보Infomation를 넣기 위해서 InfoWindow() 클래스 함수를 사용하여 내용을 정의한다.

⑦ 화면에 보여주기 위한 정보를 open() 함수를 사용해서 지도에 나타낸다.

지금까지의 내용으로 홈페이지의 기본적인 내용들을 모두 구현하였는데, 서버에 파일로 저장된 내용을 브라우저에서 보여주기만 하면 된다. 대부분의 홈페이지들은 이와 같은 원리로 만들어졌으며, 다음 장부터는 화면에서 직접적으로 보여주지는 않지만 서버에서 프로그래밍을 하는 프로그래머라면 반드시 알아야 할 데이터베이스를 사용하는 방법을 설명한다. 웹 프로그래밍은 분업화가 많이 이루어져서 다수의 개발자들이 한팀이 되어 각자 맡은 분야를 전문적으로 다루게 되는데, 이 책을 이해한 독자는 웹 프로그래밍의 전반적인 사항들을 이해하고 자신에게 필요한 사항을 파악해서 좀더 깊이있게 접근하기를 바란다.

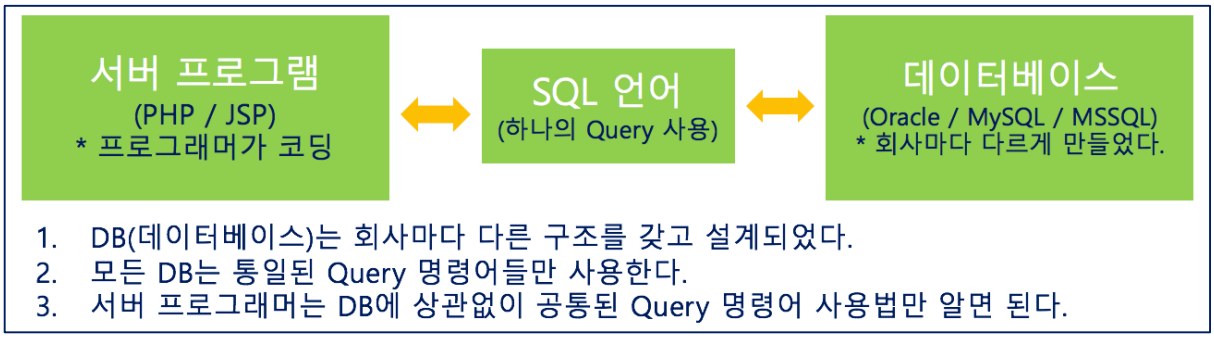
5.2 방문자 수를 데이터베이스에 저장하기

인터넷에서 정보를 제공하기 위해서는 서버를 만들어서 데이터를 보내주는 곳이 필요하다. 대부분의 웹사이트는 많은 데이터를 가지고 서비스를 하기 때문에 데이터베이스Database를 가지고 정보를 관리한다. 지금까지는 서버의 파일 시스템을 사용해서 브라우저에서 단순히 정보를 보여주었다면, 이제부터는 데이터베이스를 가지고 정보를 관리하는 방법을 사용해서 홈페이지를 만든다. 이번 장은 방문자 수를 데이터베이스에 저장하는 방법을 보여주는데, 파일에서 관리하는 것보다는 다소 복잡해 보이지만 데이터베이스의 동작원리를 이해하면 간단하게 구현할 수 있다.

[그림 5-7]은 프로그래머가 만든 서버 프로그램이 SQL 언어로 데이터베이스를 사용하는 내용이다. 데이터베이스를 이해하는 것은 데이터베이스가 어떻게 만들어졌는지는 알 필요가 없고 SQL 언어의 사용법만 알면 된다. 데이터베이스의 종류는 다양한데, 데이터베이스를 만든 회사는 자신만의 독창적인 데이터베이스를 만들었다고 이해하면 된다. 데이터베이스의 정보는 파일 구조로 데이터베이스 서버에 저장되지만,

저장된 파일에서 정보를 분리하는 방법은 회사마다 다르다. 프로그래머는 데이터베이스를 만드는 것이 아니기 때문에 데이터베이스에서 정보를 넣고 빼는 방법만을 사용하면 된다. 다양한 데이터베이스가 존재하기 때문에 프로그래머는 사용하려는 데이터베이스를 선택하고, 프로그래머가 만드는 서버의 프로그램에서 데이터베이스에 연결하는 방법을 알고, SQL 언어를 사용해서 데이터베이스를 사용하면 된다. 프로그래밍만을 공부하는 대학교의 '컴퓨터 과학과'에서는 일반적으로 4학년의 한 과목이 '데이터베이스'만을 배우는데, 데이터베이스에 정보를 저장하기 위한 테이블을 만들고 SQL을 사용해서 정보를 효과적으로 넣고 가져오는 방법을 배운다. 데이터베이스에 연결한 이후에 데이터베이스의 내용을 가져오거나 변경할 수 있는데 이 때 사용하는 명령어를 Query(쿼리)라고 말한다. 간단히 이해하자면 데이터베이스에서 사용하는 공통된 언어를 SQL이라고 하고, 명령어들을 간단히 Query라고 부른다. 이 책에서는 모든 SQL 언어를 설명하는 것이 아니기 때문에 데이터베이스 테이블을 최대한 간단히 만들고, 데이터에 접근하는 방법을 간단히 보여준다.

그림 5-7 모든 데이터베이스는 하나의 SQL 언어만을 사용한다.

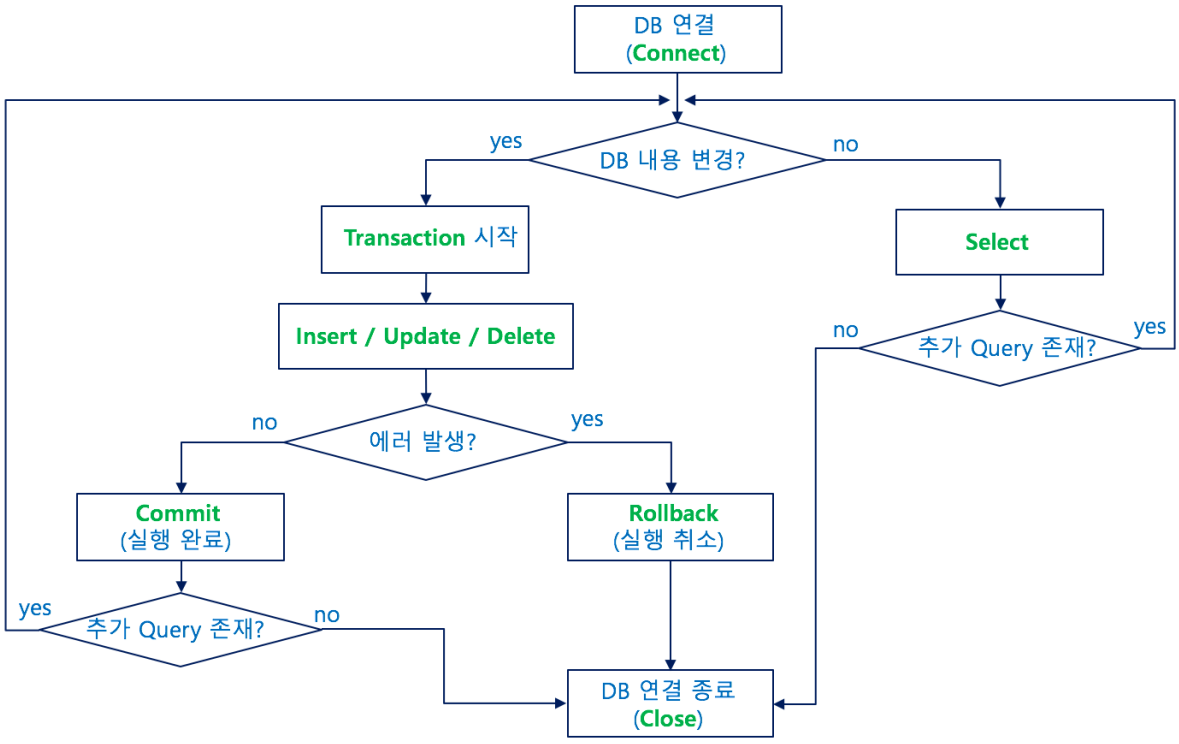


서버의 데이터베이스를 사용해서 웹 서비스를 제공하는 곳은 상당히 많은데, 예를 들면 KTX 기차표 예매, 대학교에서의 수강신청, 은행의 온라인 송금과 같이 우리가 실생활에서 사용하는 많은 서비스들이 데이터베이스를 사용한다. 데이터베이스를 처리할 때 중요하게 생각해야 할 부분은 단순한 검색을 하는 것과 데이터베이스의 내용을 변경하는 것을 구분해야 한다는 것이다. 검색은 데이터베이스의 내용을 가져오는 것이기 때문에 간단한 Query를 사용하고 속도가 상대적으로 빠르다. 은행의 온라인 서비스를 예로 들면 '계좌조회'와 같이 내용을 가져오기만 하는 것은 은행의 전산시스템에서 데이터베이스 검색만이 이루어진 것이다. 그러나, 온라인 송금과 같은 서비스는 조금 복잡하게 이루어진다. 온라인 송금 도중에 에러가 발생하면 송금하는 동안 진행된 모든 내용을 취소해야 한다. 이와 같이, 데이터베이스 처리를 하

는 도중에 발생할 수 있는 에러에 대비하기 위해서 사용하는 방법이 'Transaction'(트렌젝션)이라는 과정이다. 'Transaction'은 데이터베이스의 내용을 변경하게 되면 항상 사용하는 방법이라고 이해하면 되는데, 온라인 송금을 한 번 하게 되면 'Transaction'이 한 번 발생했다고 생각하면 된다. 'Transaction'은 데이터베이스의 내용을 변경하는 도중에 문제가 발생하면 'Transaction'이 시작하기 전 단계로 원복을 시킨다.

[그림 5-8]은 데이터베이스에 연결하고 Query를 처리하는 과정의 Flowchart를 보여주는데, 데이터베이스의 내용을 변경하는 대표적인 Query의 명령어는 추가(Insert), 변경(Update), 삭제(Delete) 등이 있다. 이와 같이 데이터베이스의 내용을 변경하는 Query가 발생할 때는 명령어의 실행 전에 'Transaction'을 시작하고 데이터베이스의 내용이 정상적으로 처리되면 완료를 위해서 'Commit'(승인)을 실행해서 'Transaction'을 정상적으로 종료하고, 만약 에러가 발생하면 'Rollback'(원복)을 실행해서 'Transaction'이 시작하기 전 상태로 원상복구 시킨다. 이 책에서는 데이터베이스를 간단히 사용하기 때문에 'Transaction'을 쉽게 코드화했지만, 실제 은행의 이체와 같은 복잡한 상황에서는 타 은행에 정상적으로 금액이 송금된 모든 과정에서 에러가 없을 경우에 'Commit'을 처리해야 하기 때문에 'Transaction'의 사용은 아주 중요하다. [그림 5-8]에서 단순한 검색(Select)을 진행하는 것은 'Transaction' 없이 데이터베이스의 내용을 가져오면 된다.

그림 5-8 SQL 언어 명령어 Flowchart

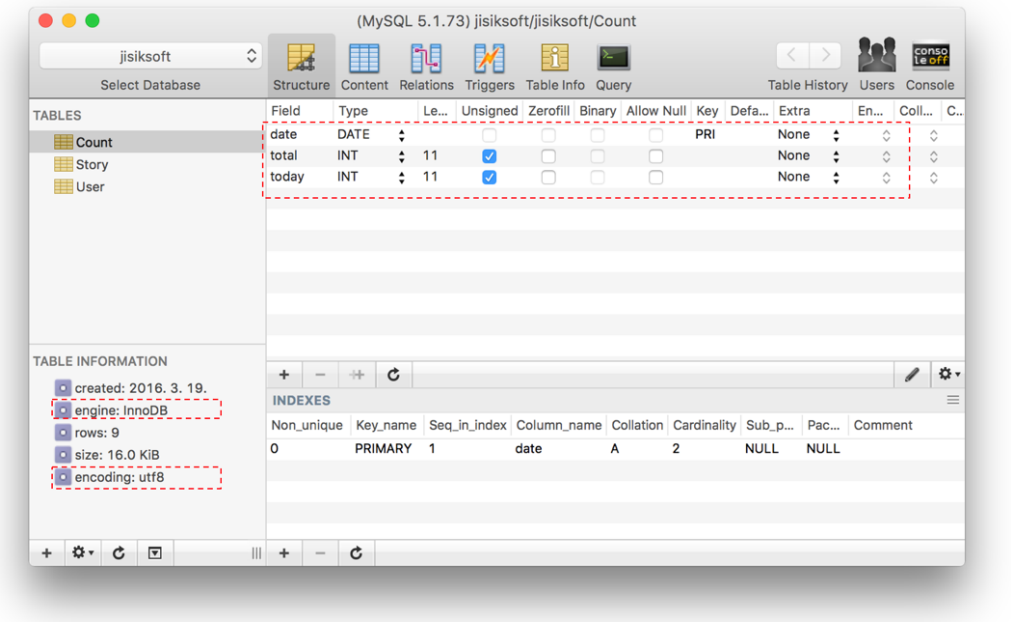


사용자가 시스템에 접속해서 데이터베이스를 사용하게 되면, 서버의 프로그램은 데이터베이스에 연결하고 Query를 처리한 후에 연결을 끊게 된다. 규모가 큰 웹 서비스를 구축할 때 가장 중요한 것이 데이터베이스를 설계하는 것인데, 웹 프로그래밍에서 가장 실력있는 사람들이 데이터베이스를 설계한다고 볼 수 있다. 만약 여러 회사가 모여서 구축하는 대형 시스템이라면 가장 핵심적인 부분을 데이터베이스로 보기 때문에 전체를 총괄하는 회사에서 가장 뛰어난 프로그래머가 데이터베이스를 설계하고 구축한다. 데이터베이스는 크게 테이블Table로 관리되는데, 같은 종류의 데이터를 하나의 테이블에서 관리한다.

[그림 5-9]는 방문자 수의 데이터를 저장하기 위해서 만든 Count 테이블의 구조를 보여주는데, 왼쪽 아래에서 중요한 것이 'InnoDB' 엔진Engine의 테이블로 만들었음을 알 수 있다. 현재는 많은 데이터베이스의 테이블들이 저장방법을 'InnoDB'로 만드는 편인데, 관계형 데이터베이스를 사용하기 위해서는 'InnoDB'로 테이블을 만드는 것이 좋다. 이 책에서는 아주 간단한 테이블들을 만들고 서로 독립적으로 사용하지만, 아주 복잡한 시스템에서는 테이블 간의 데이터들이 연관성을 가지고 운영된다. 예를 들어, 대학교의 수강신청과 같은 시스템에서 사용자들의 정보는 'User' 테이블에 저장되고 수강과목들은 'Sugang'이라는 테이블에 저장된다고 가정하면, 'Sugang' 테이블에는 'User' 테이블에 저장된 학생의 학번을 저장해서 해당 학생의 수강과목 내용을 관리해야 한다. 즉, 데이터베이스는 테이블마다 중복되어 사용해야 하는 값들을 가지고 종속된 구조를 만드는데, 이러한 구조를 "관계형 데이터베이스"라고 하고 이러한 관계형 데이터베이스를 만들 때 'InnoDB' 엔진을 사용하는 것이 가장 효과적이다. 이 책에서는 가장 간단한 데이터베이스 테이블들을 만들어서 사용하기 때문에 '관계형 데이터베이스'를 사용해서 깊이 있는 구조로 설계하지는 않았지만, 아주 간단한 데이터베이스 사용을 독자들은 경험하게 될 것이고 이후에 데이터베이스만을 설명하는 책으로 깊이 있는 데이터베이스를 접하기를 바란다.

[그림 5-9]에서 Count 테이블의 구조를 보면, date, total, today라는 필드Field들을 만들었다. 'date'는 날짜 정보를 저장하고, 'total'에는 전체 방문자 수와 'today'에는 해당 날짜의 방문자 수를 저장하기 위해서 만들었다.

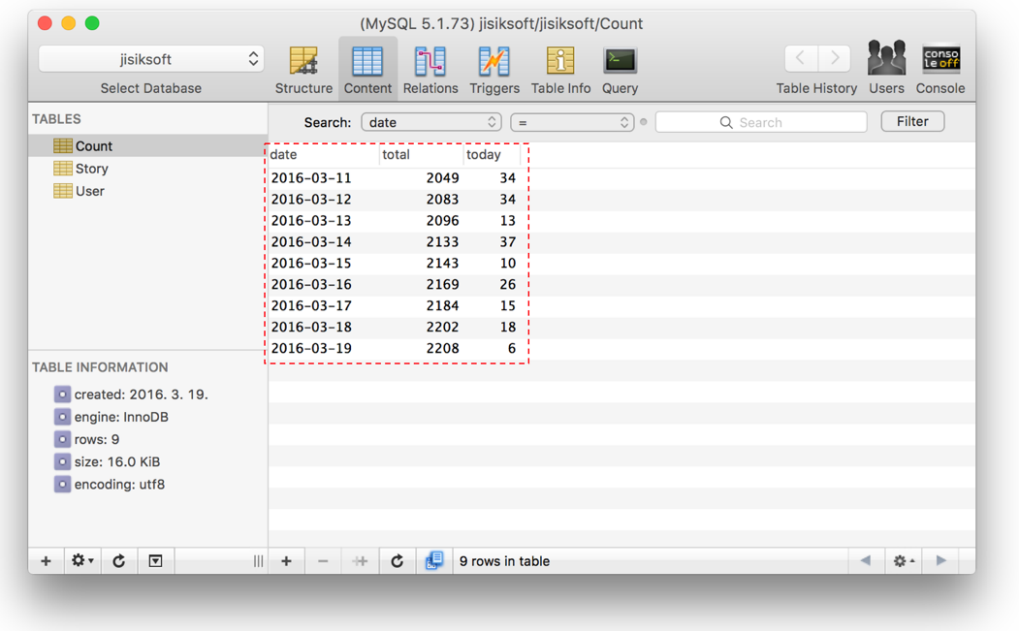
그림 5-9 방문자 수를 저장하기 위해서 만들어진 Count 테이블



데이터베이스 테이블에서 중요한 것 중의 하나가 Primary 키를 정하는 것인데, Count 테이블에서 Primary 키는 'date' 필드이다. Primary 키는 중복되지 않는 값을 가진 필드로 정해야 하는데, 테이블에서 고유한 id 값을 의미하며 각각의 저장된 데이터를 구분하기 위해서 사용된다. 데이터베이스의 데이터에 접근할 때는 Primary 키를 중심으로 검색하는 것이 좋은데, [그림 5-9]의 아래 영역에 보이는 'Indexs'에 자동으로 등록되는 Primary 키는 메모리에서 관리되기 때문에 검색의 속도가 빠르다. 실제 데이터베이스를 운용할 때에는 'Indexs'에 특정 필드들을 등록해서 사용할 수 있는데, 데이터베이스 관리자들은 아주 빈번하게 데이터베이스에 접근하는 Query가 있을 경우에 검색 조건으로 사용되는 필드들을 등록하면 데이터베이스의 처리 속도가 향상된다. 그러나, 일반적으로 데이터베이스는 아주 많은 데이터를 가지고 있기 때문에, 최소한의 필드들을 'indexs'에 등록해서 메모리에서 관리하는 데이터의 양을 적게 하는 것이 좋다. 이 책에서는 모든 테이블의 데이터는 'Primary'만을 'Index'에 등록하지만, 일반적으로 복잡한 구조의 '관계형 데이터베이스'에서는 'Foreign' 키(다른 테이블의 'Primary' 키와 같은 값을 가지는 필드)라는 것을 많이 만들어서 사용하기 때문에 'Index'에서 관리한다. 이 책에서는 간단한 데이터베이스를 만들어서 사용하기 때문에 자세한 설명은 생략한다.

[그림 5-10]에는 Count 테이블에 저장된 데이터들을 보여주는데, 'date' 필드의 값들은 날짜값으로 중복된 값이 존재하지 않는다. 이와 같이 날짜 별로 방문자 수를 저장하기 때문에 필요하다면 날짜에 따른 방문자 수를 통계로 만드는 것도 가능하다.

그림 5-10 Count 테이블에 저장된 데이터 내용

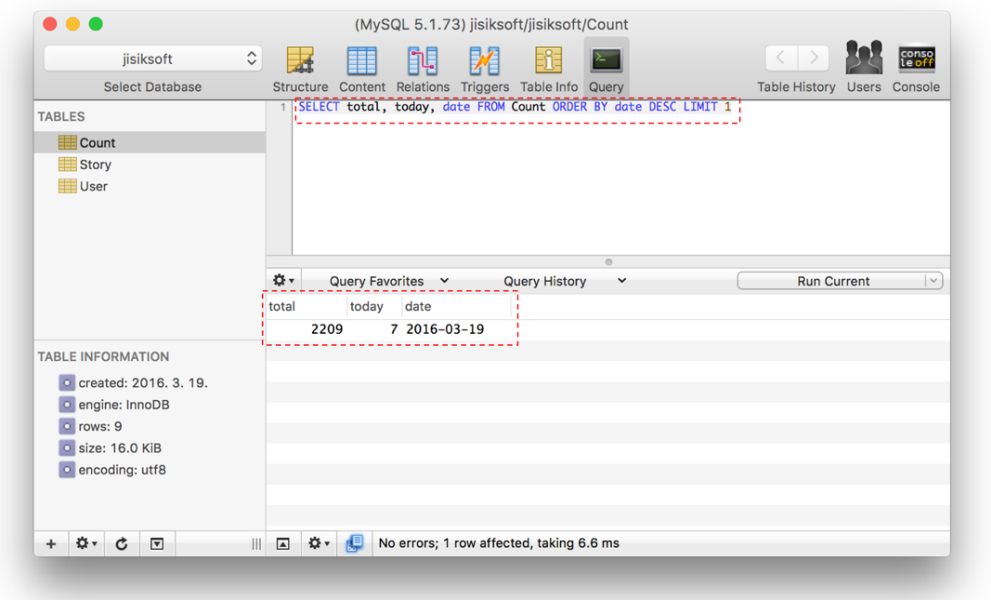


[그림 5-11]은 가장 마지막으로 저장된 날짜의 데이터를 가져오는 Query를 실행한 결과이며, Query의 명령은 아래와 같다.

SELECT total, today, date FROM Count ORDER BY date DESC LIMIT 1

위의 Query 내용을 해석하면, Count 테이블에서(FROM Count) 날짜를 내림차순으로 정렬한 후(ORDER BY date DESC) 처음 한 개(LIMIT 1)의 데이터에서 'total', 'today', 'date' 값을 가져온다.(SELECT total, today, date) 데이터베이스에서 Query를 사용한다는 것은 정해진 규칙에 따라서 명령어를 사용하면 되는데, 이와 같이 명령어에서 의미를 쉽게 알 수 있다. 프로그래밍은 영어의 알파벳을 사용해서 만드는 것이기 때문에, 다른 분야에 비해서 영어를 가장 많이 사용하는 분야라 할 수 있다. Query 명령어를 자주 사용하다보면 사용하는 예약어들이 많지 않기 때문에 생각보다 쉽게 친숙해질 수 있다.

그림 5-11 가장 최근에 저장된 날짜의 데이터를 가져오는 Query 실행 결과



데이터베이스를 사용하기 위해서는 데이터베이스에 접속해서 테이블과 필드를 만들고 Query 명령어도 실행할 수 있는 "데이터베이스 관리 소프트웨어"를 사용해야 하는데, 이 책에서는 MacOS에 설치하는 'Sequel Pro'라는 프로그램을 사용했다. 데이터베이스 관리 소프트웨어들은 종류가 많지만 기본적인 사용방법은 대부분 비슷하기 때문에 독자에게 맞는 프로그램을 선택해서 사용하면 된다.

[그림 5-12]는 '지식소프트' 홈페이지의 하단에 방문자 수를 보여주는 내용이다. 이전 Chapter에서는 방문자 수를 파일에서 관리하였지만 이번 장에서는 Count 테이블에 방문자 수를 저장해서 관리하는 방법을 설명한다. 이 책에서 사용하는 서버 프로그래밍 언어는 PHP이기 때문에 PHP에서 제공하는 예약어로 데이터베이스에 연결하고 Query를 사용하는 방법을 알수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_5/5.2/www/

그림 5-12 '지식소프트' 하단의 왼쪽에 보여지는 방문자 수

[그림 5-3]은 데이터베이스에 접속하기 위한 정보들을 관리하는 'config.php' 파일의 내용을 보여주는데, 이와 같이 데이터베이스에 접속하기 위한 설정 값들을 코드에

직접 넣는 것이 아니라 개별 파일에 저장하는 것이 시스템을 관리하는데 효과적이다. 이후에 데이터베이스 정보가 변경되면 'config.php' 파일의 정보만을 변경하면 되기 때문에 시스템 관리자가 바뀌더라도 쉽게 유지보수를 할 수 있다. 하지만 이렇게 관리하는 것은 서버가 블랙 해커에게 공격당해서 해커가 자유롭게 시스템을 조작하게 되면 가장 중요한 데이터베이스의 접속방법도 알 수 있기 때문에 위험하다. 그래서, 대형 시스템에서는 데이터베이스에 접속하는 서버(WAS 서버)들을 인터넷에서 접속할 수 없는 회사 Network의 내부망에서 관리한다. [코드 5-3]의 데이터베이스 접속 정보는 예제이며, 실제 '지식소프트' 홈페이지 서버에서는 다른 값을 사용하고 있다.

[코드 5-3] 데이터베이스에 연결하기 위한 설정 (/library/php/config.php)

```
<?php

    $dbconfig = array( 'hostname' => "localhost",                    //---①
                       'username'=> "middlestory",                   //---②
                       'password'=> "jisik123456789",                //---③
                       'database'=> "jisiksoft");                    //---④

?>
```

① 데이터베이스가 설치된 서버의 IP(인터넷 주소)를 설정하는데, 여기서는 PHP 프로그램이 동작하는 서버에 데이터베이스가 설치되었기 때문에 같은 서버를 의미하는 'localhost'로 서버 주소를 설정하였다.

② 서버에 접속하는 사용자를 지정할 수 있으며, 데이터베이스 관리자가 접근할 수 있는 사용자를 등록할 수 있으며 여기서는 middlestory 계정의 사용자가 데이터베이스에 등록되었다.

③ 데이터베이스에 접속하기 위해서는 데이터베이스에 등록된 사용자의 암호를 사용해야 한다.

④ 서버에 많은 데이터베이스를 만들 수 있으며, '지식소프트' 홈페이지는 'jisiksoft'라는 데이터베이스를 사용한다.

[코드 5-4]는 서버에 설치된 MySQL 데이터베이스를 사용하기 위해서 만들어진 PHP 클래스이며, 이와 같이 클래스로 만들어서 데이터베이스와의 연결과 Query의 요청을 단순화시키는 방법은 대부분의 프로그램에서 사용하고 있다. 프로그래밍에서 클

래스의 사용은 한 번 만들어진 클래스의 재사용이 용이하기 때문이며, 이와 같이 만들어진 클래스는 PHP와 MySQL을 사용하는 시스템이라면 항상 복사해서 사용하면 개발시간이 상당히 단축되는 장점이 있다. [코드 5-4]는 MySQL 서버에 연결하고, Query를 요청하고 받은 결과의 내용을 가져오는 함수들을 PHP 클래스로 만든 것이며, 함수 안에서는 데이터베이스와의 연동을 위해서 PHP 언어의 예약어를 사용하고 있다. 프로그래밍을 조금 단순화시키면, '라이브러리', 'API', '클래스' 등을 개별적인 개념으로 이해하는 것보다는 복잡한 프로그래밍 구조를 함수로 만들어서 단순화시킨 후에 프로그래밍을 좀더 쉽게 하는 것이 중요한 목적이라고 이해하면 모두가 같은 개념으로 받아들일 수 있다.

[코드 5-4] MySQL 데이터베이스를 사용하기 위해서 만들어진 PHP 클래스 (/library/php/mysql.php)

```
<?php

class MySQL {                                                          //---①

    var $mConn;                                                        //---②
    var $mConnected = false;                                           //---③

    function connect($data) {                                          //---④
        $host       = $data['hostname'];
        $username   = $data['username'];
        $password   = $data['password'];
        $database   = $data['database'];

        $this->mConn = mysqli_connect($host, $username, $password, $database);     //---⑤

        if (!$this->mConn){
            return false;
        }

        mysqli_set_charset($this->mConn,'utf8');                       //---⑥
        $this->mConnected = true;

        return true;
    }

    function close() {                                                 //---⑦
        if($this->mConnected == true) {
            if(!mysqli_close($this->mConn)){
                $this->mConnected = false;
```

```
                }
            }
        }

        function query($sql) {                                          //---⑧
            if($this->mConnected == true) {
                $result = mysqli_query($this->mConn, $sql);
                return $result;
            }
            return false;
        }

        function fetchArray($result) {                                  //---⑨
            if($this->mConnected == true) {
                $resultArray = array();
                while($data = mysqli_fetch_assoc($result)){
                    array_push($resultArray,$data);
                }
                return $resultArray;
            }
            return false;
        }

        function fetchRow($result) {                                    //---⑩
            if($this->mConnected == true) {
                return mysqli_fetch_assoc($result);
            }
            return false;
        }

        function startTransaction() {                                   //---⑪
            return $this->query("START TRANSACTION");
        }

        function commit() {                                             //---⑫
            return $this->query("COMMIT");
        }

        function rollback() {                                           //---⑬
            return $this->query("ROLLBACK");
        }
}

?>
```

① PHP 프로그램에서 데이터베이스의 사용을 쉽게 하기 위해서 MySQL 클래스를 만들었다. 다른 클래스 프로그래밍 언어와 마찬가지로 이후에 'new MySQL();'과 같

이 'new'를 사용해서 클래스 객체를 생성해서 사용한다.

② MySQL 데이터베이스에 연결을 한 이후에 연결된 객체를 저장하기 위해서 $mConn 변수를 사용한다. $mConn 변수가 값을 가지고 있다는 것은 데이터베이스에 연결되어 있음을 의미하며, 이 객체를 통해서 데이터베이스에 Query를 요청하고 결과를 받는다.

③ 데이터베이스에 연결이 되었는지를 확인하기 위해서 $mConnected 변수를 사용하며, MySQL 객체가 생성되어도 데이터베이스와 연결이 아직 이루어지지 않았기 때문에 초기값을 'false'로 설정한다. 이후에 데이터베이스와 연결이 이루어지면 $mConnected 변수는 'true' 값을 갖는다.

④ 데이터베이스에 연결하는 함수로서 'config.php' 파일에 저장된 데이터베이스 정보를 매개변수로 받아서 PHP 언어에서 제공하는 'mysqli_connect()' 함수를 사용해서 MySQL 데이터베이스에 연결한다.

⑤ MySQL 데이터베이스에 연결하기 위해서 'mysqli_connect()' 함수를 사용하며 정상적으로 연결이 이루어지면 현재 클래스의 객체인 $this의 멤버인 $mConn 변수는 데이터베이스와 연결된 객체Object를 저장한다. 만약, 데이터베이스와 연결이 이루어지지 않으면 '$this-mConn' 변수는 'false' 값을 갖게 되고 함수를 종료한다. PHP의 클래스 사용에서 객체의 멤버 변수에 접근하는 방법은 '$this->mConn'와 같이 'mConn' 앞에는 '$' 문자를 넣지 않는다.

⑥ 데이터베이스는 'utf8'로 테이블들이 만들어졌기 때문에 데이터베이스와 주고받는 데이터들을 'utf8'로 셋팅해야 한다.

⑦ 데이터베이스를 사용하기 위해서는 연결한 후에 Query들을 주고받으며 모든 동작이 이루어지고 난 이후에는 데이터베이스와의 연결을 종료해야 한다. MySQL 클래스에서는 'close()' 함수를 사용해서 연결을 종료하는데, PHP 언어에서 정의된 'mysqli_close()' 함수를 사용해서 MySQL 데이터베이스와의 연결을 끊는다.

⑧ 데이터베이스에 Query 명령어를 전달해서 내용을 변경하거나 검색할 수 있는데, 'mysqli_query()' 함수를 사용해서 $mConn 변수에 저장된 객체로 Query 명령어를 전달하기 위해서 query() 함수를 만들었다. 'query()' 함수는 전달하는 Query 명령을 문자열 형태인 $sql 매개변수로 받아서 $mConn 변수의 객체로 전달하고 데이터베이스로부터 받은 결과를 반환한다.

⑨ 데이터베이스로부터 전달받은 내용은 하나의 데이터 정보일 수도 있고, 다수의 데이터를 배열로 받기도 한다. 데이터베이스로부터 받은 결과를 분리해내는 과정이

필요한데, 배열 형태의 데이터를 PHP의 배열로 만들어주기 위해서 'fetchArray()' 함수를 만들었다. MySQL 데이터베이스로부터 받은 배열을 하나씩 가져오는 함수가 'mysqli_fetch_assoc()'이며, fetchArray() 함수에서는 데이터베이스의 결과인 배열의 데이터를 하나씩 가져와서 $resultArray라는 PHP 배열에 추가하고 결과를 반환한다.

⑩ 만약 데이터베이스로부터 받은 결과가 하나의 데이터라면 'mysqli_fetch_assoc()' 함수를 한 번만 사용해서 결과를 반환한다. 위에서 정의한 'fetchArray()' 함수만을 사용해서 데이터를 분리할 수도 있지만, 데이터베이스의 결과값으로 하나의 데이터만 갖는다는 것을 안다면 'fetchRow()' 함수를 사용해서 코드를 간단히 만들는 것이 좋다.

⑪ 데이터베이스의 내용을 변경할 때 사용하는 방법이 트렌젝션Transaction을 사용하는 것인데, 이와 같이 startTransaction() 함수를 만들어서 코드를 단순화시켰다.

⑫ 트렌젝션이 시작되고 Query에서 에러가 없으면 commit() 함수를 사용해서 모든 동작을 완료한다.

⑬ 데이터베이스의 내용을 변경하는데 중간에 에러가 발생하면 '트렌젝션'이 시작되기 전의 상태로 원복시키기 위해서 rollback() 함수를 사용한다.

[코드 5-5]는 서버의 파일을 내용을 읽어서 사용자 브라우저로 전송하는 'ajax_get_data.php' 파일이다. 홈페이지의 아랫부분인 'Footer 영역'의 데이터를 읽은 후에 방문자 수를 추가하는 코드가 추가되었다. 방문자 수는 데이터베이스에 저장되었기 때문에 데이터베이스에서 가장 마지막 날짜의 데이터를 읽은 후에 날짜가 프로그램이 실행되는 날짜와 같으면 방문자 수를 1 증가시키고, 날짜가 다르면 방문자 수가 1인 현재의 날짜로 데이터베이스에 추가한다. 이전까지는 데이터베이스를 사용하기 위한 준비작업이었다면 아래의 코드는 데이터베이스를 사용하는 방법을 보여주며 이후에 진행되는 모든 Query의 사용은 이와 같이 진행된다.

[코드 5-5] 홈페이지의 아래에 방문자 수를 추가하는 PHP 코드 (/data/ajax_get_data.php)

```
<?php

$dataId          = (isset($_POST['dataId'])) ? $_POST['dataId'] : "";

if ($dataId == "header")
        $filename = "./header/".$dataId.".html";
else if ($dataId == "info")
        $filename = "./info/".$dataId.".html";
else if ($dataId == "footer")
```

```
        $filename = "./footer/".$dataId.".html";
else if ($dataId == "login")
        $filename = "./login/".$dataId.".html";
else if ($dataId == "register")
        $filename = "./login/".$dataId.".html";
else if ($dataId == "personalinfo")
        $filename = "./login/".$dataId.".html";
else
        $filename = "./container/".$dataId.".html";

$fp = fopen($filename, "r");
$data = fread($fp, filesize($filename));
fclose($fp);

if ($dataId == "footer") {                                          //---①

        require_once('../library/php/config.php');                  //---②
        require_once('../library/php/mysql.php');

        $today = date("Y-m-d");                                     //---③

        $db = new MySQL();                                          //---④

        if(!$db->connect($dbconfig)) {                              //---⑤
            print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
            exit();
        }

        $sql = "SELECT total, today, date FROM Count ORDER BY date DESC LIMIT 1";   //---⑥

        $result = $db->query($sql);                                 //---⑦
        if($result == false){
            $db->close();
            print '{ "error" : "데이터베이스 테이블에서 제목을 불러오지 못했습니다." }';
            exit();
        }
        $arrCount = $db->fetchRow($result);                         //---⑧

        $db->startTransaction();                                    //---⑨

        if ($arrCount['date'] == $today) {                          //---⑩
            $sql = "UPDATE Count SET total=total+1, today=today+1 WHERE date='".$today."'";
        } else {
            $sql = "INSERT INTO Count VALUES ('".$today."', ".($arrCount['total']+1).", 1)";
```

```
    }

    $result = $db->query($sql);                                          //---⑪
    if($result == false){
        $db->rollback();                                                 //---⑫
        $db->close();
        print '{ "error" : "데이터베이스 테이블에서 을 방문자 수를 업데이트하지
                            못했습니다." }';
        exit();
    }

    $db->commit();                                                       //---⑬

    $db->close();

    $data = '<div class="color2"
style="position:relative;left:10px;width:100px;height:40px;float:left;">
            Today : <span class="color5">'
            .($arrCount['date'] == $today ? ($arrCount['today']+1) : 1).
            '</span></br>
            Total   : <span class="color5">'
            .($arrCount['total']+1).
            '</span></div>'
            .$data;                                                      //---⑭
}

print $data;
exit();

?>
```

① 서버에 'Footer 영역'의 데이터를 보낸다면 데이터베이스에 저장된 방문자 수를 추가하는 프로세스를 수행한다.

② 데이터베이스를 사용하기 위해서 'config.php' 파일에 저장된 데이터베이스 정보와 'mysql.php' 파일에서 만들어진 MySQL 클래스를 연결한다.

③ 오늘 날짜를 $today 변수에 저장한다. 만약 2016년 3월 22일이라면 $today 변수는 '2016-03-22' 값을 갖는다.

④ 데이터베이스에 연결하기 위해서 만들어진 MySQL 클래스를 'new'를 사용해서 생성한다.

⑤ 'config.php' 파일의 $dbconfig에 정의된 데이터베이스 정보를 가지고 MySQL 데이터베이스에 연결한다. 만약 연결이 이루어지지 않았으면 에러를 브라우저로 보내

고 프로그램을 종료한다.

⑥ 가장 최근에 저장된 데이터를 가져오기 위해서 Query를 문자열로 만든다. Query의 내용은 Count 테이블에서(FROM Count) 날짜를 내림차순으로 정렬하고(ORDER BY date DESC) 첫 번째 한 개의 데이터에서(LIMIT 1) '전체 방문자 수', '오늘 방문자 수', '날짜' 정보를 가져온다.(SELECT total, today, date)

⑦ Query 문자열을 데이터베이스로 전송한다. 만약, 데이터베이스로부터 정상적인 결과를 받지 못하면 데이터베이스를 종료하고 프로그램을 종료한다.

⑧ 데이터베이스로부터 받은 결과는 배열이 아닌 하나의 데이터 정보이기 때문에 fetchRow() 함수를 사용해서 결과를 가져온다.

⑨ 데이터베이스의 내용을 변경하는 Query가 수행되기 때문에 Transaction을 시작한다.

⑩ 데이터베이스로부터 가져온 결과의 날짜가 오늘이면 '전체 방문자 수'와 '오늘 방문자 수'의 값을 1 증가시키는 Query 문자열을 만들고, 오늘 날짜가 데이터베이스에 저장되지 않았다면 '오늘 방문자 수'를 1로 하는 새로운 데이터를 삽입하는 Query 문자열을 만든다.

⑪ 데이터베이스의 내용을 변경하는 Query 명령어를 데이터베이스에 전송하고 결과를 받는다.

⑫ 데이터베이스의 내용이 정상적으로 변경되지 않았으면 rollback() 함수를 사용해서 Transaction이 시작되기 전 단계로 원복시키고 프로그램을 종료한다.

⑬ 데이터베이스의 내용이 정상적으로 변경되었기 때문에 Transaction을 승인Commit하고 데이터베이스를 종료Close한다.

⑭ 'Footer 영역'의 데이터 앞에 방문자 수를 추가한다. '($arrCount['date'] == $today ? ($arrCount['today'] + 1) : 1)' 코드는 "A ? B : C" 연산자를 사용해서 데이터베이스에서 읽어온 날짜가 오늘이면 데이터베이스에서 가져온 '오늘 방문자 수'(Today)에 1을 증가시키고, 그렇지 않으면 '오늘 방문자 수'(Today)는 1로 초기화된다.

위 코드에서 보면 데이터베이스를 사용하는 방법은 의외로 짧은 코드들로 만들어졌으며, $sql 변수에 저장되는 Query 문자열를 잘 만드는 것이 중요하다. 데이터베이스를 사용하다보면 Query 문자열을 코드에서 만드는 것보다는 '데이터베이스 관리 소프트웨어'에서 Query 명령어를 만들어서 실행한 후에 정상적인 Query 명령어임

을 확인하고 코드에 적용하는 것이 편리하다. 이 책에서는 독립된 테이블들을 사용하기 때문에 복잡한 Query 명령어들을 사용하지 않지만, 데이터베이스만을 다루는 책들이 많이 있기 때문에 이 책에서는 기본적인 내용을 이해하고 테이블들이 종속되는 예제에서 사용하는 다소 어려운 Query 명령어들의 사용법을 데이터베이스 전문 서적에서 접해보기를 바란다. '데이터베이스 관리 소프트웨어' 중에서는 데이터베이스 테이블의 관계도를 만들고 데이터베이스에 자동으로 넣어주는 기능을 제공하기도 하는데, 대형 시스템에서는 테이블의 관계도를 관리해야 하기 때문에 사용하는 소프트웨어가 상당히 중요하다. 만약, 실무에서 데이터베이스를 관리해야 한다면 회사마다 선호하는 '데이터베이스 관리 소프트웨어'들이 있기 때문에 회사에서 사용하는 소프트웨어의 사용법을 익히는 것이 중요하다. 하지만, 이 책에서와 같이 간단한 데이터베이스를 만드는 것이라면 '데이터베이스 관리 소프트웨어'에서 직접 테이블을 만들어서 사용하는 것이 편리할 수 있다.

5.3 Login 만들기

이번 장에서는 대부분의 사이트에서 제공하는 '로그인'을 구현하는데, '로그인'을 이해하기 위해서는 세션Session(사용자 PC와 서버가 연결된 상태)이라는 것을 먼저 이해해야 한다. 인터넷에서 사이트의 내용을 가져올 때는 요청한 페이지를 서버가 전달하는 역할만 하기 때문에, 사용자 PC는 웹 사이트를 제공하는 서버와 연결되어 데이터를 받은 후에 즉시 연결이 끊어지게 된다. 서버로부터 데이터를 가져올 수 있도록 연결이 항상 이루어져야 한다. 사용자 PC와 서버가 연결된 상태를 말하는 세션이라는 것은 서버에서 데이터를 가지고 온 이후에도 연결이 계속 유지되는 상태를 의미한다. 쉽게 이해하자면 '로그인'을 한 이후에는 서버와의 연결이 계속 유지되는 것이고, '로그아웃'을 실행하면 연결이 끊어지게 된다. 간혹, 사용자가 '로그인'을 한 이후에 '로그아웃' 없이 브라우저를 닫고 다시 브라우저를 실행해서 사이트에 접속하더라도 '로그인' 상태가 유지되어 있는 것을 볼수 있는데, 브라우저가 닫히더라도 사용자 PC와 서버는 계속 연결을 유지하고 있었음을 알 수 있다. 다수의 사용자에게 서비스를 제공하고 있는 서버들은 영원히 세션을 유지할 수 없기 때문에 타임아웃Timeout 시간을 정해서 일정시간동안 데이터를 주고받는 행위가 없으면 연결을 자

동으로 끊게 된다. 모든 서버는 세션으로 연결할 수 있는 갯수를 정해 놓았기 때문에 새로운 연결을 하기 위해서 동작하지 않는 세션들을 일정시간 이후에 끊어야 다수의 사용자에게 원활한 서비스를 제공할 수 있다. 정의된 세션 갯수보다 많은 사용자가 접속을 시도하게 되면 연결이 되지 않는 경우가 발생하며, 사용자는 해당 사이트의 서비스를 이용할 수 없게 된다. 서버에 과도한 트래픽Traffic이 발생하는 경우에는 서버로부터 데이터를 받지 못하기 때문에 사용자 브라우저에서는 하얀 페이지가 보일 때가 있다. 세션 관리는 사이트에서 중요하게 다루어지는데, 보안이 중요한 은행에서는 사용자가 10분동안 아무런 동작을 하지 않으면 세션이 종료되며 사용자는 자동으로 '로그아웃'된 현상을 보게 된다. 또한 은행서비스의 브라우저 창을 닫으면 세션이 자동으로 끊어지는데 은행 서비스에서는 브라우저 창이 열려있는 동안 보안 프로그램이 작동하고 있음을 알수 있다.

이번 장에서는 사용자가 웹 사이트에서 '회원가입'을 하고 '로그인'을 하는 방법을 설명하는데, 사용자 정보는 데이터베이스에 저장된다. 예전에는 사용자를 구분하기 위해서 id를 많이 사용했는데, 간단한 홈페이지에서는 사용자의 이메일을 가지고 관리하는 것이 좋다. 홈페이지를 운영하다보면 간혹 자동으로 회원가입을 하는 프로그램을 만들어서 사이트를 공격하는 못된 사람들이 있는데, 회원가입을 할 때 임의로 생성되는 문자를 넣어야 진행이 되는 사이트를 만드는 이유는 이러한 악성 해커의 공격을 막기 위해서다. 여기서 공개하는 '지식소프트' 홈페이지는 '회원가입'과 '로그인'을 할 수 있는 페이지를 가지고 있지만, 회원이 필요한 서비스가 아니기 때문에 실 서비스에서는 '회원가입'을 받지 않는다. 이메일을 회원의 id로 사용하는 것은 Apple 회사에서 사용하는 정책인데, 이메일은 전세계에서 고유한 값을 가지고 있다. 우리나라와 같이 id를 가지고 운영하게 되면 글로벌 서비스에서는 중복되는 id가 많을 수 있기 때문에 좋은 방법이 아니다. 또한, 이메일로 임시 비밀번호를 보내서 가짜 이메일로 회원가입 하는 사용자를 막을 수 있는데, 많은 사이트에서는 이메일에서 승인(Confirm) 버튼으로 정상적인 이메일을 확인하지만 이번 장에서 사용하는 방법은 임시 비밀번호를 보내고 홈페이지에 '로그인' 한 후에 '비밀번호'를 변경하도록 구현하였다.

[그림 5-13]은 '로그인' 페이지를 보여주는데, 이메일과 비밀번호로 로그인을 하게되며 로그인이 이루어지면 세션Session이 연결된다.

확인 사이트: http://jisiksoft.com/book/4/chapter_5/5.3/www/

그림 5-13 로그인 페이지

이전 장에서는 MySQL 데이터베이스를 사용하기 위해서 MySQL 클래스를 만들었는데, 이번 장에서는 세션을 위해서 Session 클래스를 만들어서 사용한다. 세션의 사용도 PHP 언어에서 만들어 놓은 함수를 사용하는데, 두 가지를 이해하면 된다. 첫 번째는 세션을 시작하기 위해서 session_start() 함수를 사용하고, 두 번째는 PHP에서 이미 정의된 '$_SESSION' 이라는 전역변수의 사용이다. $_SESSION은 배열로 만들어졌으며 사용자의 모든 정보를 $_SESSION 배열에 저장할 수 있다.

[코드 5-6]은 Session 클래스를 정의한 'session.php' 파일의 내용이다. 클래스가 정의되기 전에 'session_start()' 함수를 사용해서 세션을 항상 시작하고 사용자의 응답이 30분동안 없으면 세션을 종료하기 위해서 세션에 연결된 사용자가 마지막으로 세션에 접속한 시간을 저장하기 위해서 "$_SESSION['last_activity']"를 사용하는 코드를 추가하였다. 이와 같이 'session.php' 파일에서 세션을 항상 시작하고 타임아웃을 체크하기 때문에, 세션을 사용하기 위해서 'session.php' 파일을 PHP 코드에 추가하면 세션의 기본적인 동작은 항상 실행된다. 사용자가 '로그인'을 하게되면 세션이 연결된 것이며, 세션에 사용자 정보등을 저장하기 위해서 $_SESSION 변수를 사용하였다. 세션은 Linux 운영체제에서 관리하고 PHP 언어는 세션을 사용할수 있는 함수와 변수를 제공하기 때문에 서버 프로그래밍을 하는 우리들은 PHP 언어에서의 사용법만을 알면 된다. 서버에서 프로그래밍을 하는 것은 연산자로 연산을 하는 것 이외에

도 이와 같이 데이터베이스와 세션의 사용법을 이해해야 하며 JSP를 사용하더라도 이 책에서 다루는 과정과 거의 같다고 이해하면 된다.

[코드 5-6] 세션 연결을 위해 만든 Session 클래스 (/library/php/session.php)

```php
<?php

    session_start();                                                    //---①

    if (isset($_SESSION['last_activity']) && (time() - $_SESSION['last_activity'] > 1800)) {//---②
        session_unset();
        session_destroy();
    }
    $_SESSION['last_activity'] = time();

    class Session {                                                     //---③

        function create($data){                                         //---④
            $_SESSION['login']    = "y";
            $_SESSION['email']    = $data['email'];          //email
            $_SESSION['name']     = $data['name'];                  //이름
            $_SESSION['cell']     = $data['cell'];           //전화번호
            $_SESSION['addr']           = $data['addr'];             //주소
            $_SESSION['auth']= $data['auth'];              //권한
            $_SESSION['date_reg']       = $data['date_reg'];       //등록일
            $_SESSION['date_rec']       = $data['date_rec'];       //최근접속일

            return true;
        }

        function is_login() {                                           //---⑤
            if(isset($_SESSION['login']) && ($_SESSION['login'] == "y")) {
                return true;
            } else {
                return false;
            }
        }

        function get_data_all(){                                        //---⑥
            return $_SESSION;
        }

        function get_email(){                                           //---⑦
```

```
            return $_SESSION['email'];
        }

        function set_cell($cell) {                                          //---⑧
            $_SESSION['cell'] = $cell;
        }

        function set_addr($addr) {                                          //---⑨
            $_SESSION['addr'] = $addr;
        }

        function destroy(){                                                 //---⑩
            $_SESSION = array();
            session_destroy();
            return true;
        }
    }

?>
```

① 세션을 시작한다. 'session.php' 파일을 사용할 때마다 세션은 항상 자동적으로 동작한다.

② 세션 타임아웃TimeOut을 30분으로 설정하기 위해서 추가된 코드이며, 사용자가 서버로부터 데이터를 가져가기 위해서 요청할 때마다 $_SESSION['last_activity']에는 현재의 시간이 저장된다.("$_SESSION['last_activity'] = time();") 'time()' 함수는 현재의 시간을 초단위로 알려주는 함수이며, 현재의 시간에서 이전에 접속한 시간을 뺀 값이 1800(=60초X30분) 이상이면 세션의 내용을 session_unset() 함수를 사용해서 지우고 세션의 연결을 session_destroy() 함수를 사용해서 종료한다.

③ 프로그램에서 사용하기 위한 Session 클래스이며 시스템마다 저장하는 사용자의 정보 항목들이 다르기 때문에 다른 시스템에서 Session 클래스를 사용한다면 $_SESSION 변수에 저장되는 항목을 해당 시스템에 맞게 변경해주어야 한다. 또한, '지식소프트' 홈페이지에서는 사용자 정보를 많이 사용하지 않기 때문에 Session 클래스에서 정의된 함수들이 많지 않지만, 세션을 사용하면서 사용자의 정보를 다양하게 처리하는 시스템이라면 클래스 안에 유용한 함수들을 추가로 만들어서 사용하는 것이 좋다.

④ 사용자가 로그인을 하게 되면 실행하는 함수가 session_create() 함수이며, 데이터베이스에 저장된 사용자의 정보를 $data라는 매개변수로 전달해서 세션 정보를 만들어준다. 사용자 정보를 저장하는 $_SESSION 변수의 항목들은 데이터베이스에 저장되는 사용자 정보들이며, $_SESSION['login'] 항목은 '로그인'이 되었음을 확인하

기 위해서 사용된다.

⑤ '로그인' 상태인지 아닌지에 대한 확인은 자주 발생하기 때문에 is_login() 함수를 사용해서 세션이 연결된 상태인지를 확인한다. $_SESSION['login'] 변수에 값이 존재하고 해당 값이 'y'이면 세션이 연결된 상태이기 때문에 true를 반환한다.

⑥ 세션의 저장된 모든 데이터를 전달한다. 'get_data_all()' 함수는 세션에 저장된 모든 데이터가 $_SESSION 변수에 있기 때문에 $_SESSION 변수만을 반환하고 종료한다.

⑦ 시스템에서 id로 사용되는 것이 'email'이며, get_email() 함수를 사용해서 세션에 저장된 사용자의 이메일 정보를 받는다.

⑧ 사용자가 개인정보를 변경하게 되면 set_cell() 함수를 사용해서 세션에 저장된 핸드폰 번호를 변경한다.

⑨ 사용자가 개인정보를 변경하게 되면 set_addr() 함수를 사용해서 세션에 저장된 주소를 변경한다. 이와 같이 세션의 특정 정보를 변경하는 함수를 만들어서 사용하는 것은 클래스에서 자주 사용하는 방법이다.

⑩ 세션을 종료하기 위해서 destroy() 함수를 사용하는데, 만약 사용자가 '로그아웃'을 실행하면 destroy() 함수가 실행된다. 세션의 종료는 $_SESSION 배열변수에 저장된 모든 내용을 초기화시키고, session_destroy() 함수를 사용해서 세션의 연결을 끊는다.

'지식소프트' 홈페이지에서 사용자가 페이지를 계속 보고 있다는 것은 사용자가 서버의 데이터를 요청하는 이벤트가 발생하는 것으로 알 수 있는데 사용자에게 항상 보내지는 데이터는 'Container 영역'의 데이터이기 때문에 'Container 영역'의 데이터를 보내기 전에 세션의 타임아웃을 체크하는 것이 좋다.

[코드 5-7]은 사용자 브라우저로 파일의 내용을 보내는 'ajax_get_data.php' 파일의 내용이다. 세션의 타임아웃을 체크하기 위해서 'session.php' 파일을 'Container 영역'의 데이터를 보내기 전에 추가했다. 이와 같이 'session.php' 파일에 세션을 위해서 항상 동작해야 하는 코드를 만들어서 파일의 추가만으로 세션이 항상 동작하게 하는 것은 좋은 방법이다. 물론, 세션을 확인하기 위해서 개별적으로 필요한 위치에 코드를 삽입해도 되지만, 홈페이지의 구조를 알고 있으면 필요한 위치에서 일괄적으로 적용시키는 것이 좋다.

[코드 5-7] 'Container'의 데이터 요청 시 세션의 타임아웃을 항상 체크한다. (/data/ajax_get_data.php)

```php
<?php

$dataId = (isset($_POST['dataId'])) ? $_POST['dataId'] : "";

if ($dataId == "header")
        $filename = "./header/".$dataId.".html";
else if ($dataId == "info")
        $filename = "./info/".$dataId.".html";
else if ($dataId == "footer")
        $filename = "./footer/".$dataId.".html";
else if ($dataId == "login")
        $filename = "./login/".$dataId.".html";
else if ($dataId == "register")
        $filename = "./login/".$dataId.".html";
else if ($dataId == "personalinfo")
        $filename = "./login/".$dataId.".html";
else {
        require_once('../library/php/session.php');                    //---①
        $filename = "./container/".$dataId.".html";
}

$fp = fopen($filename, "r");
$data = fread($fp, filesize($filename));
fclose($fp);

##### 생략 #####
```

---

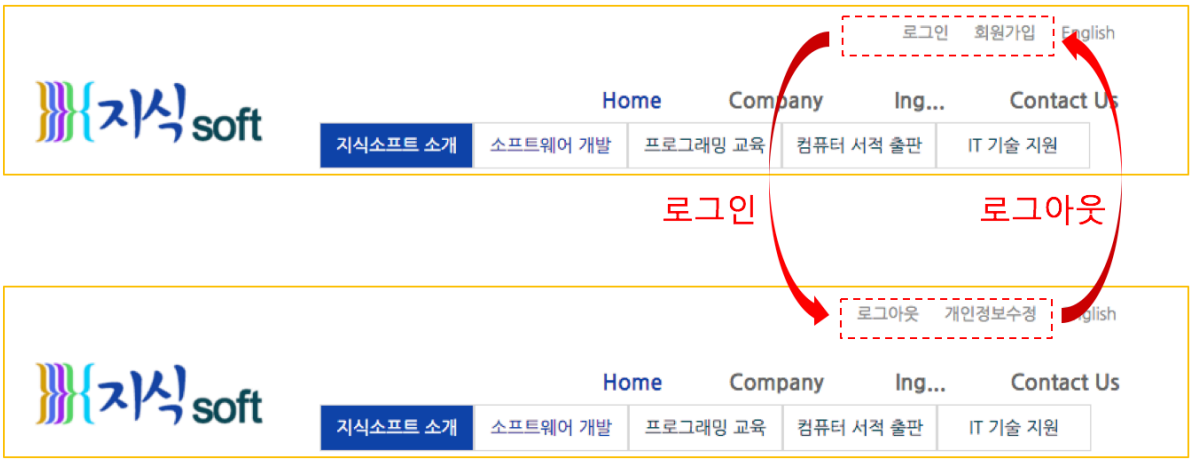
① 사용자가 홈페이지를 계속 보고 있다면 내용이 항상 변경되는 'Container 영역'의 데이터를 요청하는 것이기 때문에 'Container 영역'의 데이터를 보내기 전에 세션의 시간을 업데이트하기 위해서 'session.php' 파일을 연결해서 파일의 코드를 자동으로 실행시켜주면 된다. PHP 언어에서 'require_once()' 함수는 C/C++ 언어의 '#include'와 같은 의미로 이해하면 된다.

대부분의 홈페이지에서는 '로그인' 상태와 '로그아웃' 상태를 페이지의 상단에서 알려준다.

[그림 5-14]는 '지식소프트' 홈페이지의 상단 우측에서 보여주는 '로그인'과 '로그아웃'의 변경된 화면이다. 메뉴를 만들 때와 같은 방법으로 '로그인' 상태의 태그와 '로그아웃' 상태의 태그를 화면에서 보여주거나 안보이게 CSS 디자인을 처리하면 된다. 즉, 모든 태그는 이미 만들어져 있으며, 'display' 속성을 사용해서 화면에 출력하는 태그들의 id 값을 사용해서 CSS 디자인을 변경한다. [그림 5-14]의 상단에는 '로그아웃' 상태에서 '로그인'과 '회원가입'의 사용자 메뉴가 보이는 그림이며, 하단에는 '로그인'이 정상적으로 진행된 이후에 '로그아웃'과 '개인정보수정'의 사용자 메뉴가 보

이는 그림이다.

그림 5-14 '로그인'과 '로그아웃' 상태에서 따라서 '사용자 메뉴'가 변경된다.



[코드 5-8]은 홈페이지 상단의 'Header 영역'에서 사용자 메뉴의 내용을 만드는 HTML 코드와 사용자 메뉴의 항목이 클릭되었을 때의 이벤트 처리를 위한 JavaScript 코드를 보여준다. 지금까지는 JavaScript 코드를 '.js' 파일에서 관리했지만, [코드 5-8]의 JavaScript는 홈페이지 전체에서 사용하는 것이 아니라 'Header 영역'에서만 사용하는 함수들이기 때문에 'header.html' 파일 안에서 구현하였다. 이와 같이 특정 페이지에서만 동작하는 JavaScript 코드는 '.html'에서 추가하는 것이 프로그램을 좀더 체계적으로 관리하는데 좋다. 실생활에서 규모가 큰 시스템은 다수의 사용자가 개발하고 관리하게 되는데, 몇 년 동안 문제없이 사용하다가 시스템의 유지보수를 하는 업체가 변경되고 새로운 기능을 추가하게 되면 시스템의 프로그래밍 코드에 일관성이 없어지는 경우가 발생하곤 한다. 프로그래머의 입장에서는 다른 사람이 만들어 놓은 코드를 분석하는 것도 힘들 수 있기에 자기만의 방식대로 추가 코드를 구현하기도 하는데, 한참이 지난 후에는 전체 시스템의 코드를 분석하는 것이 힘들어진다. 이를 두고 '스파게티 코드'라고 부르곤 한다. 스파게티처럼 꼬여있어서 정상적으로 동작은 하지만 이해하기가 어렵다는 의미이며 효율성도 떨어지는 코드를 말한다.

[코드 5-8] Header 영역의 사용자 메뉴의 HTML 과 JavaScript 코드 (/kr/data/header/header.html)

```
<div id="logo_jisiksoft" onclick="clickedSubMenu('smenu_1_1');">
        <img src="./img/logo_jisiksoft.png">
</div>
```

```
<div id="header_user">
    <div class="user" id="login" onclick="clickedLogin();">로그인</div>            //---①
    <div class="user" id="logout" onclick="clickedLogout();">로그아웃</div>
    <div class="user" id="register" onclick="clickedRegister();">회원가입</div>
    <div class="user" id="personalinfo" onclick="clickedPersonalInfo();">개인정보수정</div>
    <div class="user" onclick="location.href='../'">English</div>
</div>

##### Header 영역의 메뉴 코드 생략 #####

<script>

    if (isLogedin() == 1) {                                                        //---②
        changeLogedin();
    } else {
        changeLogedout();
    }

    function isLogedin() {                                                         //---③
        var result = doAjax('./data/login/ajax_is_login.php');
        return result; //1:login , 0:not login
    }

    function clickedLogout() {                                                     //---④

        var result = doAjax('./data/login/ajax_logout.php');

        if (result == 1) {
                changeLogedout();
                clickedSubMenu('smenu_2_1');
        }
    }

    function clickedLogin() {                                                      //---⑤
        var param = { dataId:"login" };
        var result = doAjax('./data/ajax_get_data.php', param);
        $("#container").empty().html(result);
    }

    function clickedRegister() {                                                   //---⑥
        var param = { dataId:"register" };
        var result = doAjax('./data/ajax_get_data.php', param);
        $("#container").empty().html(result);
    }
```

```
function clickedPersonalInfo() {                                    //---⑦
    var param = { dataId:"personalinfo" };
    var result = doAjax('./data/ajax_get_data.php', param);
    $("#container").empty().html(result);
}

function changeLogedin() {                                          //---⑧
        $("#login").css({ "display": "none" });
    $("#logout").css({ "display": "block" });
        $("#register").css({ "display": "none" });
    $("#personalinfo").css({ "display": "block" });
}

function changeLogedout() {                                         //---⑨
        $("#login").css({ "display": "block" });
    $("#logout").css({ "display": "none" });
        $("#register").css({ "display": "block" });
    $("#personalinfo").css({ "display": "none" });
}

</script>
```

① 사용자 메뉴에 보여지는 내용을 모두 태그로 만들었으며, 태그의 id 값을 가지고 객체를 화면에 보여주거나 사라지게 디자인을 변경한다. 각각의 태그 객체는 클릭했을 때 이벤트가 발생하고 JavaScript 함수를 실행한다.

② 페이지가 브라우저에서 열리면 isLogedin() 함수가 자동으로 실행되어 '로그인' 상태인지를 확인하고, '로그인' 상태이면 사용자 메뉴에 '로그아웃'과 '개인정보수정' 텍스트를 보여주고 '로그아웃' 상태이면 사용자 메뉴에 '로그인'과 '회원가입' 텍스트를 보여준다.

③ '로그인' 상태인지를 확인하는 함수로서 서버의 'ajax_is_login.php' 파일을 실행하고, 서버로부터 받은 결과가 1이면 '로그인' 상태이고 0이면 '로그인'이 되지 않은 상태이다.

④ 사용자 메뉴에서 '로그아웃'이 클릭되면 실행되는 함수로서, '로그아웃'을 처리하기 위해서 서버의 'ajax_logout.php' 파일을 실행한다. 서버에서 '로그아웃'이 정상적으로 이루어지면 결과값으로 1을 받고 changeLogedout() 함수를 실행해서 사용자 메뉴의 텍스트를 '로그아웃' 상태로 변경하고 'Container 영역'에 새로운 페이지를 불러온다.

⑤ 사용자 메뉴에서 '로그인'이 클릭되면 실행되는 함수로서 'Container 영역'에서 로그인 페이지를 열기 위해 서버로부터 'login.html' 파일의 내용을 가져온다.

⑥ 사용자 메뉴에서 '회원가입'이 클릭되면 실행되는 함수로서 서버에 있는 'register.html' 파일의 내용을 받아서 'Container 영역'에 출력한다.
⑦ 사용자 메뉴에서 '개인정보수정'이 클릭되면 실행되는 함수로서 서버로부터 'personalinfo.html' 파일의 내용을 가져와서 화면에 보여준다.
⑧ '로그인' 상태의 사용자 메뉴를 보여주는 함수로서 '로그인'과 '회원가입' 메뉴를 사라지게 하고, '로그아웃'과 '개인정보수정' 메뉴를 화면에 보여준다.
⑨ '로그아웃' 상태의 사용자 메뉴를 보여주는 함수로서 바로 위의 changeLogedin() 함수와 반대의 동작을 수행한다. 즉, '로그아웃' 상태에서는 사용자 메뉴에 '로그인'과 '회원가입' 텍스트가 보인다.

'로그인'이 된다는 것은 세션이 연결된다는 것을 의미하기 때문에 로그인 페이지에서 서버로 '이메일'과 '비밀번호'를 전달하면 서버는 데이터베이스에 있는 회원 정보와 비교하여 세션을 생성하게 된다. '로그인' 과정은 이후에 '로그인' 페이지에서 구체적으로 설명하며 여기서는 세션이 이루어진 상태에서 세션이 연결되었는지를 확인하는 방법과 세션의 연결을 종료하는 방법을 설명한다.
[코드 5-9]는 세션이 연결되었는지를 확인하는 'ajax_is_login.php' 파일의 코드를 보여주며, [코드 5-10]은 '로그아웃'이 실행될 때 세션을 종료하는 'ajax_logout.php' 파일의 코드를 보여준다.

[코드 5-9] 로그인이 되었는지 확인하는 PHP 코드 (/data/login/ajax_is_login.php)

```
<?php

require_once('../../library/php/session.php');                    //---①

$ses = new Session();                                              //---②

if($ses->is_login()){                                              //---③
        print 1;
} else {
        print 0;
}

exit();

?>
```

① 세션 클래스가 정의된 'session.php' 파일을 연결해서 코드를 가져온다.

② Session 클래스를 생성하고, 생성된 객체를 $ses 변수에 저장한다.

③ Session 클래스의 is_login() 함수를 실행해서 세션이 연결되었으면 1을 반환하고, 아니면 0을 사용자 브라우저로 보낸다.

[코드 5-10] '로그아웃' 요청을 처리하는 서버의 PHP 코드 (/data/login/ajax_logout.php)

```
<?php

require_once('../../library/php/session.php');

$ses = new Session();

if($ses->destroy()) {                                                    //---①
        print 1;
} else {
        print 0;
}

exit();

?>
```

① Session 클래스의 destroy() 함수를 실행해서 세션의 연결을 종료하고, 정상적으로 세션이 종료되었다면 사용자 브라우저로 1을 보낸다.

이번 장에서의 내용은 세션을 이해하는 것이 중요한데, 세션이 연결된다는 것은 데이터베이스에 저장된 사용자 정보와 일치하는 사람의 접속이 정상적으로 이루어진 것이다. 즉, 사용자가 '로그인'을 서버로 요청하게 되면, 서버는 사용자의 '이메일'과 '비밀번호'가 데이터베이스에 저장된 값과 일치를 확인 후 세션을 연결한다. 지금부터는 세션을 연결하기 위해서 사용자의 정보를 저장하고 있는 데이터베이스를 다루며, User 테이블을 만들어서 관리하게 된다. 일반적으로 사용자의 '이메일', '이름', '비밀번호'만을 가지고 User 테이블을 관리할 수 있지만, 대부분의 시스템에서는 사용자의 '주소'와 '전화번호'도 함께 가지고 있다.
[그림 5-15]는 User 테이블에 저장되는 필드를 보여주며, 아래는 각 필드의 내용을 설명한다.

'email' : 사용자의 고유한 id 값으로 사용되는 이메일 주소

'name' : 사용자 이름

'passwd' : 비밀번호로서 암호화되어 저장되기 때문에 42-bytes의 길이를 갖는다.

'cell' : 사용자의 전화번호

'addr' : 사용자의 집 주소를 문자열로 갖는다.

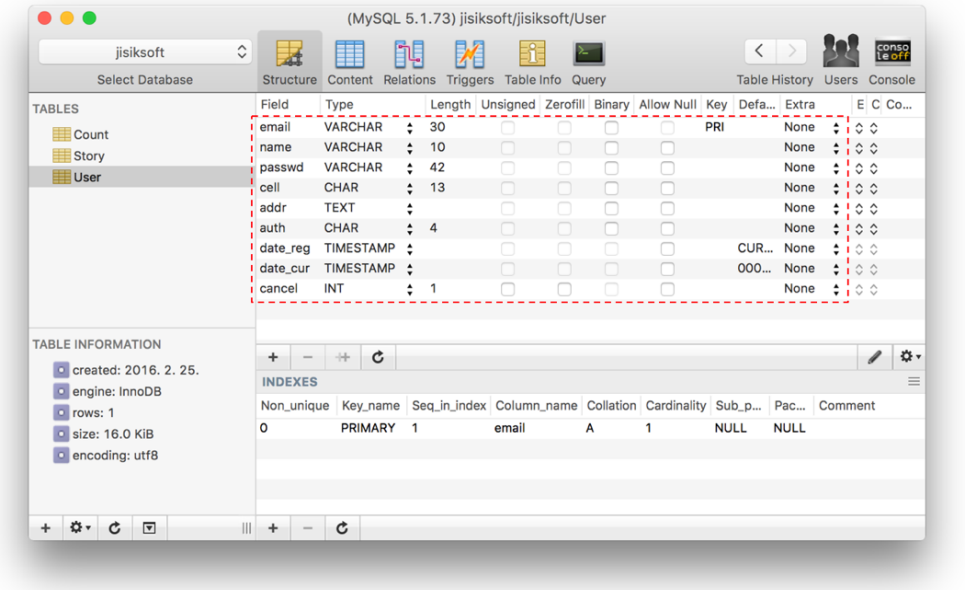
'auth' : '권한Authority'을 위해서 4-bytes의 문자열이 할당되었는데, 이 책에서는 '임시 비밀번호'가 부여되면 '0000'이라는 값을 저장하고, 정상적으로 '로그인' 후에 8자리 이상의 비밀번호를 등록하면 '0001'로 변경한다. 시스템에서 사용자의 권한을 세분화해서 관리하는 경우가 많은데, 회사의 직급과 같은 구분도 데이터베이스를 설계하면서 문자열로 관리할 수 있다. 이 책에서는 사용법을 단순화하기 위해서 정상적인 비밀번호가 등록되었는지를 확인하는 방법으로만 사용했다.

'date_reg' : 사용자가 회원가입Register을 한 날짜를 저장한다.

'date_cur' : 사용자가 마지막으로 로그인한 날짜Current Date를 저장한다.

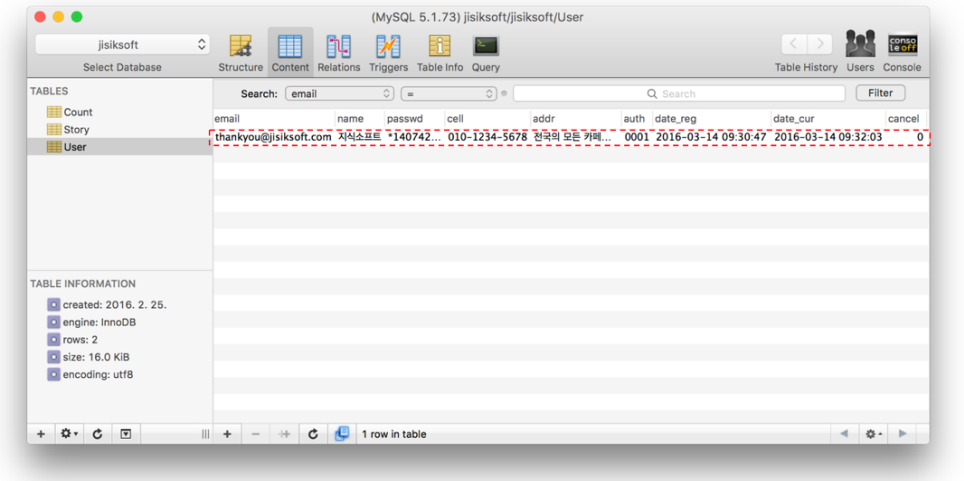
'cancel' : 정상적인 사용자는 0의 값을 갖고, '회원탈퇴'를 한 사용자는 1의 값을 갖는다. 여기서 만드는 데이터베이스는 '관계형 데이터베이스'가 아니기 때문에 사용자의 정보를 지워도 상관없지만, 일반적인 '관계형 데이터베이스'에서는 다른 테이블에 저장된 데이터에 사용자 id가 사용될 수 있기 때문에 사용자 정보를 데이터베이스에서 지우지 못하고 이와 같이 필드를 만들어서 관리할 수도 있다. 회사 시스템을 만들 때 이와 같이 처리하면 퇴사한 직원의 정보를 지우지 않고 기존 직원과 구분해서 관리할 수 있다.

그림 5-15 사용자를 저장하기 위해 만들어진 User 테이블



[그림 5-16]은 데이터베이스의 User 테이블에 데이터가 저장된 내용을 보여주는데, 저장된 사용자 정보는 '비밀번호'만을 제외하고는 모두 읽을 수 있는 내용이다. 데이터베이스에서는 대부분의 패스워드를 암호화해서 저장하는데, MySQL 데이터베이스도 password() 함수를 제공하기 때문에 데이터베이스에서 데이터를 암호화하는 것은 쉽다. 간혹 사용자의 '비밀번호'를 암호화하지 않고 저장하는 시스템도 있는데, 암호화하는 과정이 데이터베이스 서버의 CPU에 부하를 준다고 생각하기 때문이다. 그러나, 사용자의 비밀번호를 암호화하지 않는 것은 굉장히 위험하며, 만약 사용자가 이 사실을 알게 된다면 해당 사이트에 대한 신뢰는 없어진다. 만약 암호화하지 않은 비밀번호가 블랙 해커에게 탈취되면, 하나의 암호를 다수의 시스템에서 사용하는 사람들도 있기 때문에 2차 피해도 발생한다. 암호학을 공부한 사람이라면 데이터베이스에 저장되는 모든 데이터를 암호화해서 저장한다고 생각할 수도 있는데, 암호화와 복호화 과정은 서버의 CPU에 부하를 많이 주는 과정이기 때문에 거의 사용하지 않는 방법이다. 또한 암호화와 복호화 알고리즘 코드가 서버에 같이 있다면 보안을 유지할 수 없다. 데이터베이스에 저장된 정보를 보호한다는 것은 서버를 외부의 공격으로부터 안전하게 지키는 것이 가장 좋은 방법이다. 어쩌면 데이터베이스에 접근할 수 있는 사람에 대한 보안교육이 더 중요하다고 생각하는 것이 좋다.

그림 5-16 User 테이블에 사용자가 저장된 데이터 결과

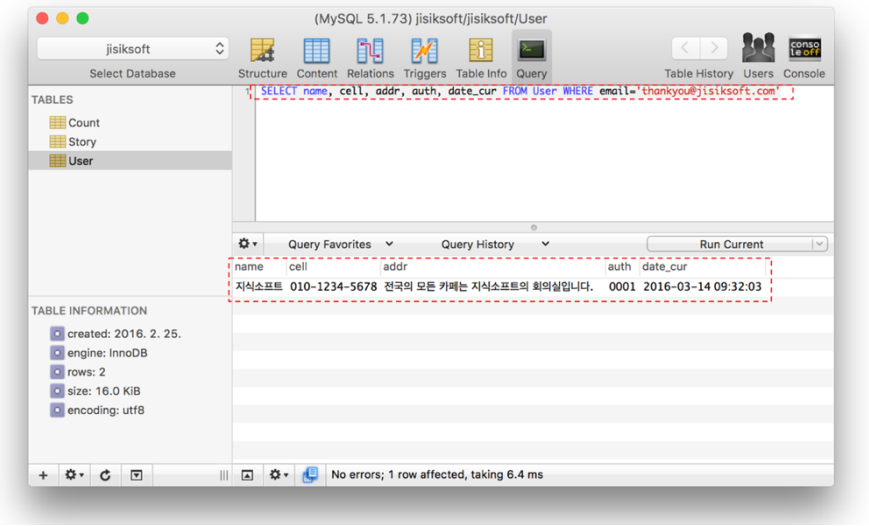


[그림 5-17]은 '이메일'로 사용자 정보를 검색하는 Query의 결과를 보여주는데, Query의 내용은 아래와 같다.

SELECT name, cel, addr, auth, date_cur FROM User WHERE email='thankyou@jisiksoft.com'

위의 Query를 해석하면, User 테이블에서(FROM User) 이메일이 'thankyou@jisiksoft.com'인(WHRER email='thankyou@jisiksoft.com') 사용자의 이름, 전화번호, 주소, 권한, 최근접속날짜 등의 정보를 검색한다.(SELECT name, cell, addr, auth, date_cur) 데이터베이스를 관리하는 사람은 데이터베이스에서 필요로하는 정보를 '데이터베이스 관리 소프트웨어'에서 Query를 직접 사용해서 필요한 정보를 가져온다. 이 책에서는 데이터베이스의 가장 간단한 운영방법들을 소개하는 것이라 생각하고 깊게 이해해야할 내용들은 데이터베이스만을 설명하는 책을 통해서 Query의 사용을 좀더 깊이 이해하기를 권한다.

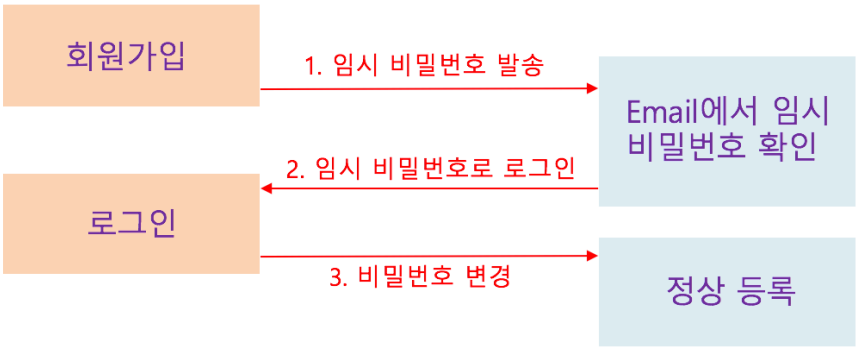
그림 5-17 이메일로 사용자 정보를 조회한 Query 결과



이제부터는 홈페이지에서 회원가입을 하고 로그인을 하는 방법을 구현하는데, [그림 5-18]은 이메일을 id로 사용하고 '임시 비밀번호'를 사용해서 정상적인 사용자를 등록하는 방법의 그림을 보여주는데 등록되지 않은 이메일로 회원가입을 하는 사용자를 방지하기 위해서 괜찮은 방법이다. 사이트를 운영하다보면 악의적으로 회원가입과 로그인을 계속 반복해서 수행하는 자동 프로그램을 만들어서 사용하는 블랙 해커들이 있다. 이러한 공격을 막기 위해서 이메일로 임시 비밀번호를 보낸 후 재접속하게 하는 방법을 사용한다. 간혹 사이트에서 잘 보이지 않는 이미지 문자를 입력하는 것을 요구하는데, 자동 프로그램의 사용을 막기 위해서 사용하는 방법일 수 있다. 이 책에서 사용하는 이메일을 사용하는 방법은 최근의 추세인데, 메일을 받을 수 있는 정상적인 이메일 계정을 사용해서 공격하는 것도 의미가 없고, 하나의 이메일은 하나의 계정을 만들 수 있기 때문에 반복적인 공격을 실행할 수가 없다. 이 책에서는 내용을 간단히 하기 위해서 만들지는 않았지만, 임시 비밀번호를

발송한 이후에 정해진 시간 안에 로그인 하지 않은 계정을 자동으로 지우는 프로그램을 서버에서 주기적으로 자동실행하는 방법을 사용하면 좋다.

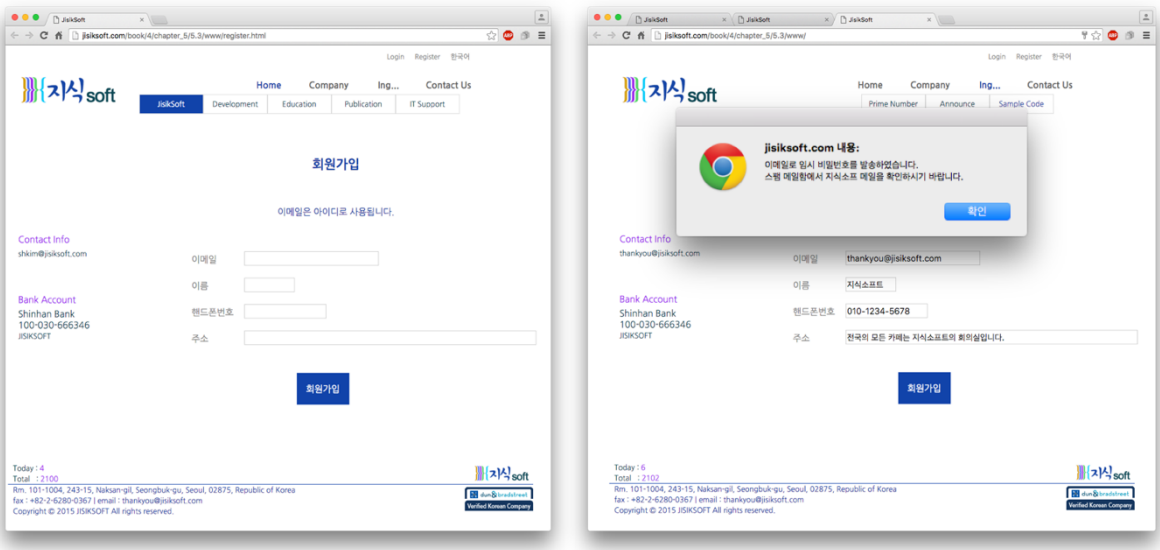
그림 5-18 임시 비밀번호를 사용한 회원가입 방법



[그림 5-19]는 '회원가입' 버튼을 클릭하고 회원가입 페이지에서 정보를 입력한 후의 결과를 보여준다. 오른쪽에 나타난 메시지 팝업은 정상적으로 회원가입이 완료되어 임시 비밀번호를 등록한 이메일로 발송했다는 내용이다. 물론, 시스템에서 등록된 이메일 주소가 정상적인지 판별하지는 못하고 단지 해당 이메일 주소로 '임시 비밀번호'를 포함한 이메일을 발송할 뿐이다. 간혹 사용자의 이메일 서비스에서 대량 발송하는 이메일을 차단하기 위해서 등록되지 않은 도메인 주소에서 발송하는 메일을 수신하지 않는 경우도 있을 수 있다. 시스템을 만들면서 '스팸 메일함'에서도 발송된 이메일을 받지 못한다면, 사용자의 이메일 서비스 회사에 이메일을 발송하는 도메인 주소를 알려주어야 한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_5/5.3/www/register.html

그림 5-19 회원가입 초기 페이지와 회원가입이 진행된 결과



[코드 5-11]은 브라우저에서 '회원가입'을 위한 페이지를 위한 HTML과 JavaScript 코드를 보여주며, [코드 5-12]는 서버에서 '회원가입'을 처리하는 PHP 코드를 보여준다. [코드 5-11]은 브라우저에서 입력한 내용을 가져와서 서버로 전송하는 단순한 코드이며, 서버로부터 받은 결과에 따라서 사용자에게 메시지 창을 보여준다. 회원가입에서의 결과는 첫째는 '로그인' 상태에서는 회원가입이 이루어지지 않으며, 둘째는 정상적으로 서버에 등록된 이후에 이메일을 발송했음을 알려주며, 마지막 메시지는 서버에서 발생한 에러를 알려준다. 회원가입의 경우에는 필요한 정보를 사용자에게 알려주면 된다.

[코드 5-11] 회원가입 페이지의 HTML 과 JavaScript 코드 (/data/login/register.html)

```
</br></br>
<div class="big bold color1" style="text-align:center;">회원가입</div>
</br></br>
<div class="color1" style="text-align:center;">이메일은 아이디로 사용됩니다.</div>
</br></br>
<div style="position:relative;left:100px;width:700px;height:400px;">
    <div class="textTitle">이메일</div>
    <input class="inputbox" type="text" id="email">                    //---①
    <div style="height:20px;clear:left"></div>
    <div class="textTitle">이름</div>
    <input class="inputbox" type="text" id="name">                     //---②
    <div style="height:20px;clear:left"></div>
    <div class="textTitle">핸드폰번호</div>
    <input class="inputbox" type="text" id="cell">                     //---③
    <div style="height:20px;clear:left"></div>
    <div class="textTitle">주소</div>
    <input class="inputbox" type="text" id="addr">                     //---④

    <div style="height:50px;clear:left"></div>
    <div id="btnRegister" onclick="runRegister();">회원가입</div>      //---⑤
</div>

<script language=javascript>

$(window).unbind("mousedown");                                         //---⑥

function runRegister() {
```

```
        var param = { email:$("#email").val(), name:$("#name").val(), cell:$("#cell").val(),
                      addr:$("#addr").val() };
        var result = doAjax('./data/login/ajax_register.php', param);                    //---⑦

        if (result == 0) {
            alert("이미 로그인 되었습니다.");
        } else if (result == 1) {                                                        //---⑧
            alert("이메일로 임시 비밀번호를 발송하였습니다.\n
                   스팸 메일함에서 지식소프트 메일을 확인하시기 바랍니다.");
            clickedLogin('login');
        } else {
            alert(result);
        }
}

</script>
```

① id 값이 'email'인 <input> 태그를 사용해서 사용자의 이메일 주소를 입력받는다.

② id 값이 'name'인 <input> 태그를 사용해서 사용자의 이름을 입력받는다.

③ id 값이 'cell'인 <input> 태그를 사용해서 사용자의 전화번호를 입력받는다.

④ id 값이 'addr'인 <input> 태그를 사용해서 사용자의 주소를 입력받는다.

⑤ '회원가입' 버튼을 클릭하면 'runRegister()' 함수를 실행한다.

⑥ 홈페이지를 마우스로 드래그하는 것을 방지하기 위해서 윈도우의 'mousedown' 이벤트를 동작하지 않게 처리했었는데, 여기서는 <input> 태그에 사용자 정보를 넣어야 하기 때문에 마우스를 사용해서 입력 창을 선택할 수 있게 하기 위해서 이전에 'mousedown' 이벤트에서 설정된 것을 해제한다.

⑦ jQuery의 val() 함수를 사용해서 사용자의 모든 정보를 가져와서 서버에 'Post 방식'으로 전달하고, 서버의 'ajax_register.php' 파일의 실행결과를 받는다.

⑧ 함수에서 받은 결과가 0이면 이미 '로그인'이 되어있는 상태이며, 1을 받으면 정상적으로 서버의 데이터베이스에 사용자 정보가 등록되고 임시 비밀번호를 사용자 이메일 주소로 보내졌다는 메시지 창을 보여주고 로그인 페이지로 이동한다. 그 이외의 결과값은 서버에서 에러가 발생한 것이므로 서버로부터 받은 에러 메시지를 브라우저에서 보여준다.

[코드 5-12]는 사용자의 회원가입을 실행하는 서버의 PHP 코드이며, 6자리의 숫자로 이루어진 임시 비밀번호를 만들어서 사용자 정보와 함께 서버의 데이터베이스에 등

록한다. 서버에 사용자 정보가 정상적으로 등록이 된 이후에는 사용자의 이메일 주소로 임시 비밀번호가 포함된 이메일을 발송한다. Linux 서버는 이메일 시스템을 가지고 있기 때문에 PHP 언어에서 제공하는 mail() 함수를 사용해서 특정 이메일 주소로 메일을 발송할 수 있다. PHP 언어의 mail() 함수를 보면, 대량으로 전송되는 이메일(스팸메일) 프로그램을 서버에서 쉽게 만들 수 있음을 알 수 있다. 이메일로 보내는 내용은 태그들을 사용해서 보기 좋게 만드는 것이 가능하다.

[코드 5-12] '회원가입' 요청을 처리하는 서버의 PHP 코드 (/data/login/ajax_register.php)

```
<?php

require_once('../../library/php/config.php');                                    //---①
require_once('../../library/php/session.php');
require_once('../../library/php/mysql.php');

$ses = new Session();                                                             //---②
$db = new MySQL();                                                                //---③

if($ses->is_login()) {                                                            //---④
        print 0;
        exit();
}

if(isset($_POST['email']))          { $email = $_POST['email']; }                 //---⑤
if(isset($_POST['name']))           { $name = $_POST['name']; }
if(isset($_POST['cell']))   { $cell = $_POST['cell']; }
if(isset($_POST['addr'])) { $addr = $_POST['addr'];          }

$passwd = rand(100000, 999999);                                                   //---⑥

$curTime = date("Y-m-d h:i:s", time());                                           //---⑦

if(!$db->connect($dbconfig)) {                                                    //---⑧
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

$sql = "SELECT COUNT(email) FROM User WHERE email = '".$email."' ";               //---⑨

$result = $db->query($sql);
if($result == false){
```

```
        $db->close();
        print '{ "error" : "데이터베이스 검색을 실패했습니다." }';
        exit();
}
$data = $db->fetchRow($result);

if ($data['COUNT(email)'] != 0) {                                    //---⑩
        $db->close();
        print '{ "error" : "이미 등록된 이메일입니다. 비밀번호 찾기를 해주시기 바랍니다." }';
        exit();
}

$db->startTransaction();                                             //---⑪

$sql = "INSERT INTO User VALUES ('".$email."', '".$name."', password('".$passwd."'), '".$cell."',
'".$addr."', '0000', '".$curTime."', '".$curTime."', 0)";            //---⑫

$result = $db->query($sql);
if($result == false){
        $db->rollback();                                             //---⑬
        $db->close();
        print '{ "error" : "데이터베이스에 등록되지 않았습니다." }';
        exit();
}

$db->commit();                                                       //---⑭

$db->close();

//======================== Email 보내기 ========================
$subject = '(지식소프트) 임시 비밀번호를 확인하시기 바랍니다.';          //---⑮

$message = "지식소프트의 회원가입에 감사합니다.

                임시 비밀번호는 아래와 같습니다.

                ".$passwd."

                로그인 후 비밀번호를 변경하시기 바랍니다.";

$headers    = "From: 지식소프트<thankyou@jisiksoft.com>"."\r\n".
                "Content-Type: text/plain;charset=utf-8\r\n".
              "X-Mailer: PHP/".phpversion();
```

```
mail($email, $subject, $message, $headers);
//============================================================

print 1;
exit();

?>
```

① 데이터베이스와 세션을 사용하기 위해서 클래스가 정의된 파일들을 연결한다.

② 세션을 사용하기 위해서 Session 클래스를 생성한다.

③ 데이터베이스를 사용하기 위해서 MySQL 클래스를 생성한다.

④ '로그인' 상태인지를 확인하기 위해서 Session 클래스의 is_login() 함수를 실행하고, '로그인' 상태이면 브라우저에 결과값으로 0을 전송하고 회원가입 프로세스를 진행하지 않는다.

⑤ 브라우저로부터 받은 사용자의 '이메일 주소', '이름', '전화번호', '주소' 값들을 'Post 방식'으로 받는다.

⑥ 6자리의 숫자로 이루어진 '임시 비밀번호'를 rand() 함수를 사용해서 임의의 숫자로 생성한다.

⑦ 데이터베이스에 등록 날짜와 최근 접속한 날짜를 저장하기 위해서 시간을 포함한 날짜 데이터를 date() 함수로 만들어서 $curTime 변수에 저장한다.

⑧ 데이터베이스에 접속한다.

⑨ 사용자 정보를 등록하기 전에 이미 데이터베이스에 등록된 이메일 주소인지를 확인한다. 그래서 브라우저로부터 받은 이메일 주소가 데이터베이스에 존재하는지를 검색한다. 검색된 내용의 갯수를 카운트하는 데이터베이스의 COUNT() 함수를 사용해서 해당 이메일이 존재하면 결과값은 1을 가져오게 되고, 존재하지 않으면 결과값은 0을 갖게 된다.

⑩ 해당 이메일 주소를 가지고 검색한 Query의 결과값에 내용이 존재하면 이미 등록된 이메일 주소이기 때문에, 사용자에게 에러 메시지를 보내고 프로그램을 종료한다.

⑪ 사용자 정보의 이메일 주소가 등록되지 않았기 때문에 데이터베이스에 사용자 정보를 추가하기 위해서 Transaction을 시작한다.

⑫ 사용자 정보를 입력하는 Query이며, User 테이블에(INTO User) 사용자 정보 값들을(VALUES 이하의 내용) 추가한다.(INSERT)

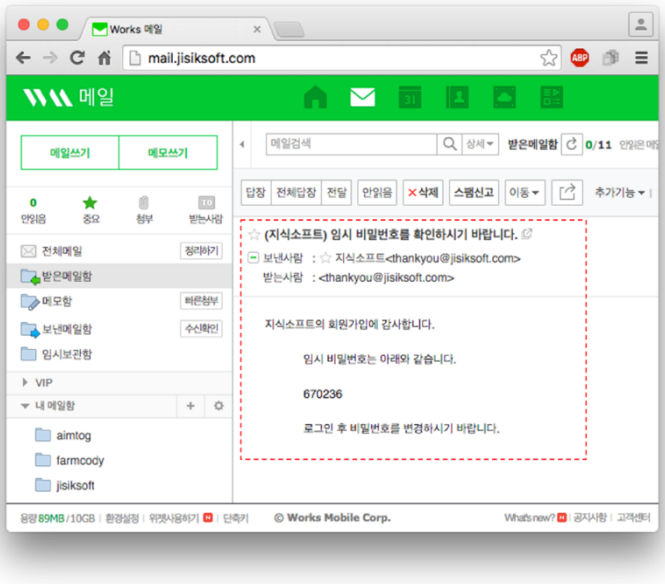
⑬ 사용자 정보가 정상적으로 등록되지 않으면, 원복(Rollback)하고 에러 내용을 브라우저로 보내고 프로세스를 종료한다.

⑭ 사용자 정보가 정상적으로 등록되면 데이터베이스의 변경된 내용들을 승인(Commit)하고, 데이터베이스와의 연결을 종료한다.

⑮ 사용자의 등록된 이메일 주소로 이메일을 보낸다. $subject 변수에는 이메일의 제목을 넣고, $message 변수에는 이메일의 내용을 넣으며, $headers 변수에는 이메일의 설정 값들을 넣는다. PHP 언어에서 제공하는 mail() 함수를 사용해서 사용자의 이메일 주소로 이메일을 발송한다. 모든 프로세스가 종료되면, 정상적으로 끝났음을 알리는 1을 브라우저로 보내고 프로그램을 종료한다.

회원가입이 정상적으로 이루어진 후에는 사용자의 이메일 주소로 메일을 발송하며, [그림 5-20]과 같이 사용자는 '임시 비밀번호'를 포함한 이메일을 받게 된다. 일반적으로 서버에서 직접 전송하는 이메일은 '스팸 메일'로 분류되어서 사용자 메일 서비스의 '스팸메일함'에 저장된다. 우리나라의 많은 사이트는 사용자에게 물건을 보내기 위해서 우편번호를 포함한 상세한 주소를 넣는 환경을 만들지만, '지식소프트'와 같이 사용자 정보가 특별히 필요하지 않는 사이트는 사용자의 전화번호와 주소가 필요하지는 않다. 실제 외국의 많은 사이트들은 사용자의 이메일 주소와 비밀번호만 데이터베이스에서 관리하는 경우가 많다.

그림 5-20 회원가입 요청으로 6 자리 숫자로 이루어진 '임시 비밀번호' 이메일을 발송한 결과



회원가입 완료 후 사용자는 '임시 비밀번호'를 가지고 시스템에 로그인한다. [그림 5-21]은 '로그인' 페이지를 보여주는데, 이메일 주소와 비밀번호를 가지고 로그인을 진행한다. 화면에는 일반적인 사이트들과 마찬가지로 '비밀번호 찾기'와 '회원가입'을

위한 항목들이 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_5/5.3/www/

그림 5-21 '로그인' 화면에서 임시 비밀번호로 사이트에 접속할 수 있다.

[코드 5-13]은 '로그인' 페이지를 만드는 HTML 코드와 이벤트를 위한 JavaScript 코드를 보여주는데, 이메일 주소와 비밀번호로 로그인을 진행하고 '비밀번호 찾기'와 '회원가입'을 아래에 추가하였다. '로그인' 과정은 서버의 'ajax_login.php' 파일을 실행시키는데 서버에 저장된 이메일 주소와 비밀번호가 같으면 세션을 연결해 준다. '비밀번호 찾기'는 'ajax_find_password.php' 파일을 실행시키는데, 이메일 주소가 등록되었으면 저장된 비밀번호를 '임시 비밀번호'로 변경하고 등록된 이메일 주소로 메일을 발송한다. 데이터베이스에 등록된 사용자가 아니면 '회원가입'을 선택해서 이전에 구현한 '회원가입' 페이지를 화면에 보여준다.

[코드 5-13] '로그인' 페이지의 HTML 과 이벤트를 처리하는 JavaScript 코드 (/data/login/login.html)

```
</br></br>
<div class="big bold color1" style="text-align:center;">로그인</div>
</br></br>
<div style="text-align:center;">지식소프트에 오신 것을 환영합니다.</div>
</br></br>
<div style="position:relative;left:100px;width:700px;height:500px;">
        <div class="textTitle">이 메 일</div>
```

```
        <input class="inputbox" type="text" id="email">                    //---①
            <div style="height:20px;clear:left"></div>
        <div class="textTitle">비밀번호</div>
        <input class="inputbox" type="password" id="passwd">               //---②
        <div style="height:70px;clear:left"></div>
        <div id="btnLogin" onclick="runLogin();">로그인</div>              //---③
        <div style="height:40px;clear:left"></div>
        <div class="color5" id="message">안녕하세요.</div>
        <div style="height:40px;clear:left"></div>
        <div style="position:relative;width:200px;float:left;">비밀번호를 잊으셨나요?</div>
        <div class="btnString" onclick="findPasswd();">비밀번호 찾기</div>  //---④
        <div style="height:10px;clear:left"></div>
        <div style="position:relative;width:200px;float:left;">아직 회원이 아닌신가요?</div>
        <div class="btnString" onclick="clickedRegister();">회원가입</div>  //---⑤
</div>

<script>

$(window).unbind("mousedown");

function runLogin() {                                                      //---⑥

    var param = { email:$("#email").val(), passwd:$("#passwd").val() };
    var result = doAjax('./data/login/ajax_login.php', param);

    if (result == 0) {                                                     //---⑦
     alert("이미 로그인 되었습니다.");
     changeLogedin();
    } else if (result == 1) {                                              //---⑧
     clickedSubMenu('smenu_2_1');
     changeLogedin();
    } else if (result == 2) {                                              //---⑨
     clickedPersonalInfo();
     changeLogedin();
    } else {                                                               //---⑩
     alert(result);
    }
}

function findPasswd() {                                                    //---⑪

    var param = { email:$("#email").val() };
```

```
    var result = doAjax('./data/login/ajax_find_password.php', param);

    if (result == 0) {
     alert("이미 로그인 되었습니다.");
    } else if (result == 1) {                                          //---⑫
     alert("이메일로 임시 비밀번호를 발송하였습니다.\n스팸 메일함에서 지식소프트 메일을 확인하시기 바랍니다.");
    } else {
     alert(result);
    }
}

</script>
```

① id 값이 'email'인 <input> 태그를 만들어서 사용자의 이메일 주소를 입력받는다.

② id 값이 'passwd'인 <input> 태그를 만들어서 사용자의 비밀번호를 입력받는다.

③ 태그로 만들어진 '로그인' 버튼을 클릭하면 runLogin() 함수를 실행해서 로그인 프로세스를 진행한다.

④ '비밀번호 찾기' 텍스트를 클릭하면 findPasswd() 함수를 실행시켜서 화면에 등록된 이메일 주소를 서버로 전달하고 '임시 비밀번호'를 이메일로 전송한다.

⑤ '회원가입' 텍스트를 클릭하면 회원가입을 위한 페이지로 이동한다.

⑥ '로그인'을 진행하는 함수로서 화면에 입력된 '이메일 주소'와 '비밀번호'를 서버로 전송하고 서버의 'ajax_login.php' 함수를 실행시킨다.

⑦ 서버로부터 받은 결과값이 0이면 이미 '로그인' 상태에 있었음을 알려준다.

⑧ 서버로부터 받은 결과값이 1이면 정상적으로 로그인 되어 세션이 연결되었기 때문에 '로그인' 페이지가 아닌 일반 페이지로 변경한다. '로그인' 상태에서는 홈페이지의 가장 위에 있는 사용자 메뉴를 chageLogedin() 함수를 실행해서 '로그인' 상태로 변경한다.

⑨ '임시 비밀번호'로 '로그인'이 이루어지면 비밀번호를 변경해야 한다. 세션이 이루어진 상태에서는 비밀번호를 쉽게 수정할 수 있게 만들었으며 '개인정보 수정' 페이지로 이동한다.

⑩ 서버에서 로그인이 진행되면서 에러가 발생하면 에러 내용을 메시지 창에 보여준다.

⑪ '비밀번호 찾기' 텍스트가 클릭되면 실행되는 함수로서, 사용자의 이메일 주소를 서버에 전달하고 서버의 'ajax_find_password.php' 파일을 실행한다. '비밀번호 찾기'는 사용자 비밀번호를 '임시 비밀번호'로 변경하고 사용자 이메일 주소로 '임시 비밀

번호'를 포함한 이메일을 발송한다.

⑫ 서버에서 수행하는 '비밀번호 찾기'가 정상적으로 끝나면 결과값으로 1을 받으며, 이때 사용자의 이메일 주소로 '임시 비밀번호'가 보냈음을 메시지 창으로 알려준다.

[코드 5-14]는 '로그인' 요청에 대한 서버에서 진행되는 과정을 구현한 PHP 코드이며, 현재 사용자가 데이터베이스에 정상적으로 등록되었는지를 확인하고 세션을 연결하는 방법을 구현했다.

[코드 5-14] '로그인' 요청을 처리하는 서버의 PHP 코드(/data/login/ajax_login.php)

```
<?php

require_once('../../library/php/config.php');
require_once('../../library/php/session.php');
require_once('../../library/php/mysql.php');

$ses = new Session();
$db = new MySQL();

if ($ses->is_login()){                                                    //---①
        print 0;
        exit();
}

if (isset($_POST['email']))   { $email = $_POST['email'];   }             //---②
if (isset($_POST['passwd']))   { $passwd = $_POST['passwd'];     }

if(!$db->connect($dbconfig)) {                                            //---③
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

$sql = "SELECT * FROM User WHERE email='".$email."' AND cancel=0";        //---④

$result = $db->query($sql);
if($result === false){
        $db->close();
        print '{ "error" : "데이터베이스 조회에 실패했습니다. (1)" }';
        exit();
}
$data = $db->fetchArray($result);

if($data == null){                                                        //---⑤
```

```
        $db->close();
        print '{ "error" : "등록되지 않은 이메일입니다." }';
        exit();
}

$sql = "SELECT * FROM User WHERE email='".$email."' AND
                passwd=password('".$passwd."')";                    //---⑥

$result = $db->query($sql);
if($result === false){
        $db->close();
        print '{ "error" : "데이터베이스 조회에 실패했습니다. (2)" }';
        exit();
}
$data = $db->fetchRow($result);

if($data == null){                                                  //---⑦
        $db->close();
        print '{ "error" : "비밀번호가 맞지 않습니다." }';
        exit();
}

$curTime = date("Y-m-d h:i:s", time());

$db->startTransaction();

$sql = "UPDATE User SET date_cur='".$curTime."' WHERE email='".$email."'";    //---⑧

$result = $db->query($sql);
if($result == false){
        $db->rollback();
        $db->close();
        print '{ "error" : "데이터베이스에 최근 접속시간을 업데이트하지 못했습니다." }';
        exit();
}

$db->commit();
$db->close();

$ses->create($data);                                                //---⑨

if (strlen($passwd) > 7)                                            //---⑩
        print 1;
else
        print 2;

exit();
```

```
?>
```

① '로그인'의 요청을 받았지만, 이미 '로그인' 상태이면 세션이 연결된 것이기 때문에 브라우저로 0을 전송하고 프로그램을 종료한다.

② 브라우저로부터 이메일 주소와 비밀번호를 'Post 방식'으로 받는다.

③ MySQL 데이터베이스에 연결한다.

④ User 테이블에 이메일 주소가 등록되었는지를 검색해서 결과를 가져온다. 'cancel' 필드의 값이 0이면 현재 정상적으로 등록된 사용자를 의미하며, 1이면 탈퇴한 사용자이기 때문에 "cancel=0"이라는 조건을 추가했다. 검색 Query의 내용은 User 테이블에서(FROM User) 'Post 방식'으로 받아서 $email 변수에 저장된 이메일 주소와 같고 탈퇴를 하지 않은 사용자(WHERE email='".$email."' AND cancel=0)의 모든 정보를 검색결과로 가져온다.(SELECT *)

⑤ 검색된 결과가 없다면 이메일 주소가 등록되지 않았기 때문에 브라우저로 에러 메시지를 전송하고 프로그램을 종료한다. 이메일 주소와 비밀번호를 동시에 Query에 넣어서 검색을 하지 않은 이유는 등록된 이메일인지, 아니면 비밀번호가 잘못되었는지를 판단하기 위해서인데 지금까지는 등록된 이메일 주소인지를 확인하는 과정이다.

⑥ 이메일이 등록되었기 때문에 비밀번호와 같이 검색해서 결과를 가져온다. 만약 결과를 가져오지 못하면 비밀번호가 잘못된 것으로 판단한다. 검색 Query의 내용은 User 테이블에서(FROM User) 이메일 주소와 비밀번호가 일치하는 사용자의(WHERE email='".$email."' AND passwd= password('".$passwd."')) 모든 정보를 검색결과로 가져온다.(SELECT *)

⑦ 검색된 결과가 없으면 비밀번호가 잘못되었기 때문에 에러 메시지를 브라우저로 보내고 프로그램을 종료한다.

⑧ 데이터베이스에 등록된 정상적인 이메일 주소와 비밀번호이기 때문에 세션을 생성해야 하는데, 데이터베이스의 연결을 종료하기 전에 최근에 로그인한 시간을 업데이트해야 한다. 업데이트를 위한 Query의 내용은 User 테이블을 업데이트하는데(UPDATE User) 이메일 주소가 $email 변수에 저장된 것과 같은(WHERE email='".$email."') 사용자의 최근 접속 날짜를 현재 날짜로 설정한다.(SET date_cur='".$curTime."')

⑨ 사용자와의 세션을 연결하기 위해서 Session 클래스의 create() 함수를 사용해서 세션을 생성한다. 세션의 생성은 create() 함수의 매개변수를 사용해서 사용자 정보

를 Session 클래스에 저장하는 것으로 간단히 이루어진다.

⑩ 임시 비밀번호는 6자리의 숫자이고 정상적인 비밀번호는 8자리 이상의 숫자이기 때문에, 임시 비밀번호로 로그인한 경우에는 사용자 브라우저에서 '개인정보수정' 페이지로 이동해야 하기 때문에 1을 전달하고 그렇지 않은 경우에는 2를 전달한다.

[코드 5-15]는 '비밀번호 찾기'를 처리하는 서버의 PHP 코드이며, '회원가입'을 처리하는 [코드 5-12]와 비슷한 과정을 진행하는데 '임시 비밀번호'를 생성한 후 데이터베이스의 비밀번호를 변경하는 과정이 다르다.

[코드 5-15] '비밀번호 찾기' 요청을 처리하는 서버의 PHP 코드 (/data/login/ajax_find_password.php)

```
<?php

require_once('../../library/php/config.php');
require_once('../../library/php/session.php');
require_once('../../library/php/mysql.php');

$ses = new Session();
$db = new MySQL();

if($ses->is_login()){
        print 0;
        exit();
}

if(isset($_POST['email']))          { $email = $_POST['email']; }

$passwd = rand(100000, 999999);

if(!$db->connect($dbconfig)) {
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

$sql = "SELECT COUNT(email) FROM User WHERE email = '".$email."' ";          //---①

$result = $db->query($sql);
if($result == false){
      $db->close();
        print '{ "error" : "데이터베이스에서 이메일 정보를 가져오지 못했습니다." }';
        exit();
}
$data = $db->fetchRow($result);

if ($data['COUNT(email)'] == 0) {
```

```
        $db->close();
        print '{ "error" : "등록되지 않은 이메일입니다." }';
        exit();
}

$db->startTransaction();

$sql = "UPDATE User SET passwd=password('".$passwd."') WHERE email='".$email."'"; //---②

$result = $db->query($sql);
if($result == false){
        $db->rollback();
        $db->close();
        print '{ "error" : "데이터베이스에 임시패스워드가 등록되지 않았습니다." }';
        exit();
}

$db->commit();

$db->close();

//==================== Email 보내기 ======================

$subject = '(지식소프트) 임시 비밀번호를 확인하시기 바랍니다.';                    //---③

$message = "비밀번호를 초기화 했습니다.
                    임시 비밀번호는 아래와 같습니다.
                    ".$passwd."
                    로그인 후 비밀번호를 변경하시기 바랍니다.";

$headers    = "From: 지식소프트<thankyou@jisiksoft.com>"."\r\n".
                    "Content-Type: text/plain;charset=utf-8\r\n".
                "X-Mailer: PHP/".phpversion();

mail($email, $subject, $message, $headers);
//=====================================================

print 1;
exit();

?>
```

① 이메일 주소가 데이터베이스에 있는지를 확인하는 Query로서 등록되지 않은 이메일 주소이면 에러 메시지를 전송하고, 등록된 이메일 주소이면 이후에 비밀번호를 변경하는 프로세스를 수행한다. Query의 내용은 User 테이블에서(FROM User) 브라우저로부터 받은 이메일 주소와 일치하는(WHERE email='".$email."') 사용자의 이메

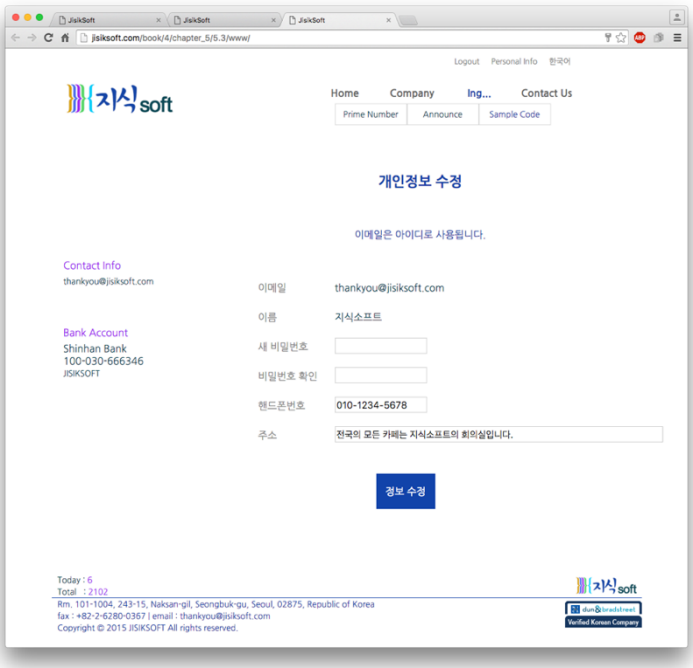
일의 갯수를 카운트해서 가져온다.(SELECT COUNT(email))

② 사용자의 비밀번호를 6자리의 숫자로 이루어진 '임시 비밀번호'로 변경한다. Query의 내용은 User 테이블을 업데이트하는데(UPDATE User) 브라우저로부터 받은 이메일 주소와 일치하는(WHERE email='".$email."') 사용자의 비밀번호를 변경한다.(SET passwd=password('".$passwd."'))

③ 사용자의 이메일 주소로 변경된 비밀번호를 포함하는 이메일을 mail() 함수를 사용해서 발송한다.

'비밀번호 변경'은 아주 중요하기 때문에 많은 사이트에서 '개인정보 수정' 페이지에서 이전 비밀번호를 다시 한번 입력하게 만드는 경우가 있다. 세션이 이루어진 상황에서는 정상적으로 등록된 사용자이기 때문에 굳이 다시 비밀번호를 재입력할 필요가 없다. 어쩌면 PC방과 같은 곳에서 브라우저로 '로그아웃'을 하지 않고 나왔을 때, 다른 사람이 이미 연결되 세션으로 비밀번호를 변경해서 계정을 뺏는 위험성을 차단하기 위해서 비밀번호를 다시 입력하게 한다고 볼 수 있다. '임시 비밀번호'로 세션이 이루어진 이후에는 사용자 고유의 비밀번호로 변경해야 하는데, 이 책에서는 '개인정보 수정' 페이지에서 비밀번호를 변경할 수 있게 만들었다. [그림 5-22]는 개인정보를 변경할 수 있는 페이지를 보여준다.

그림 5-22 '비밀번호' 변경을 하기 위한 '개인정보 수정' 페이지



개인정보를 수정하는 페이지에는 비밀번호를 제외한 사용자의 정보를 페이지에서 보여주어야 하는데, 서버의 세션에 저장된 개인정보를 가져와서 화면에 출력하면 된다. [코드 5-16]은 '개인정보 수정' 페이지의 HTML 코드와 이벤트를 위한 JavaScript 코드를 보여주는데, '회원가입' 페이지의 HTML 코드와 비슷하며 페이지의 코드를 받아온 이후에 자동으로 실행되는 JavaScript 코드는 서버로부터 사용자 정보를 가져와서 출력을 위한 태그에 내용을 넣는 과정을 진행한다.

[코드 5-16] '개인정보 수정'의 HTML과 이벤트를 위한 JavaScript 코드 (/data/login/personalinfo.html)

```
</br></br>
<div class="big bold color1" style="text-align:center;">개인정보 수정</div>
</br></br>
<div class="color1" style="text-align:center;">이메일은 아이디로 사용됩니다.</div>
</br></br>

<div style="position:relative;left:100px;width:700px;height:500px;">
    <div class="textTitle">이메일</div>
    <div class="inputbox" type="text" id="email"></div>                    //---①
    <div style="height:20px;clear:left"></div>
    <div class="textTitle">이름</div>
    <div class="inputbox" type="text" id="name"></div>
    <div style="height:20px;clear:left"></div>
    <div class="textTitle">새 비밀번호</div>
    <input class="inputbox" type="password" id="passwd">                   //---②
    <div style="height:20px;clear:left"></div>
    <div class="textTitle">비밀번호 확인</div>
    <input class="inputbox" type="password" id="passwd2">
    <div style="height:20px;clear:left"></div>
    <div class="textTitle">핸드폰번호</div>
    <input class="inputbox" type="text" id="cell">
    <div style="height:20px;clear:left"></div>
    <div class="textTitle">주소</div>
    <input class="inputbox" type="text" id="addr">

    <div style="height:50px;clear:left"></div>
    <div id="btnEdit" onclick="runPersonalInfo();">정보 수정</div>          //---③
</div>

<script language=javascript>

$(window).unbind("mousedown");
```

```
var result = doAjax('./data/login/ajax_get_user_info.php');                    //---④

if (result == 2) {                                                             //---⑤
        alert("로그인을 먼저 해주시기 바랍니다.");
}

var data = JSON.parse(result);                                                 //---⑥

$("#email").empty().html(data["email"]);                                       //---⑦
$("#name").empty().html(data["name"]);
$("#cell").attr("value",data['cell']);
$("#addr").attr("value",data['addr']);

function runPersonalInfo() {                                                   //---⑧

        var passwd = $("#passwd").val();

        if (passwd.length < 8) {                                               //---⑨
            alert("비밀번호는 8자리 이상이어야 합니다.");
            return false;
        }
        if (passwd != $("#passwd2").val()) {                                   //---⑩
            alert("'새 비밀번호'와 '비밀번호 확인'이 같지 않습니다.");
            return false;
        }

       var param = { passwd:$("#passwd").val(), cell:$("#cell").val(), addr:$("#addr").val() };
       var result = doAjax('./data/login/ajax_personal_info.php', param);      //---⑪
        alert(result);

        var result = doAjax('./data/login/ajax_get_user_info.php');            //---⑫
        var data = JSON.parse(result);
        $("#email").empty().html(data["email"]);
        $("#name").empty().html(data["name"]);
        $("#cell").attr("value",data['cell']);
        $("#addr").attr("value",data['addr']);
}

</script>
```

---

① '개인정보 수정' 페이지에서 이메일 주소와 이름은 변경이 불가능하기 때문에 <input> 태그가 아닌 태그를 사용해서 이메일 주소와 이름을 화면에 보여준다. 클래스 이름을 'inputbox'로 정한 것은 다른 개인정보와 같은 디자인을 갖

게 하기 위해서이다.

② 비밀번호를 포함한 핸드폰번호와 주소는 수정이 가능하기 때문에 <input> 태그를 사용해서 내용을 변경할 수 있게 구성하였다.

③ '정보 수정' 버튼을 클릭하면 runPersonalInfo() 함수를 실행해서 수정된 개인정보를 서버에 업데이트한다.

④ 페이지의 코드가 브라우저에 다운받아지면 실행되는 JavaScript 코드로서 사용자 정보를 서버로부터 가져와서 화면에 보여주기 위해서 서버의 'ajax_get_user_info.php' 파일을 실행한다.

⑤ '로그인'이 되지 않으면 '개인정보 수정' 페이지를 열 수 없지만, 만약 세션이 끊겼을 수 있기 때문에 세션이 연결되지 않았다면 서버로부터 2를 받기 때문에 로그인을 위한 메시지를 사용자에게 보여준다.

⑥ 서버로부터 받은 결과는 JSON 형식의 문자열 데이터이기 때문에, 문자열을 JSON 데이터로 변환하기 위해서 JavaScript 언어에서 제공하는 JSON.parse() 함수를 사용한다.

⑦ 서버로부터 받은 개인정보의 내용을 화면에 넣어준다. 정보를 넣기 위한 태그의 객체를 가져오기 위해서 jQuery를 사용했는데, 태그에 넣는 내용을 넣는 방법과 <input> 태그의 객체에 값을 넣는 방법이 다르다. 즉, 태그에서는 객체의 HTML 코드 내용을 변경하고, <input> 태그에서는 값Value을 변경해서 화면에 개인정보를 출력한다.

⑧ 수정된 개인정보를 서버로 전송해서 업데이트하는 함수이며, 비밀번호가 8자리 이상이고 확인을 위한 두 번째 비밀번호와 일치하는지를 확인한 후에 서버로 결과를 전송한다.

⑨ 비밀번호의 문자열 길이를 확인한 후에 8자리 이하이면 에러 메시지를 메시지창으로 보여준다.

⑩ 비밀번호는 화면에서 읽을 수 없기 위해서 확인을 위해서 두 번 입력하게 만드는데, 두 개의 비밀번호가 일치하지 않으면 에러 메시지를 메시지 창으로 보여준다.

⑪ 변경된 개인정보의 내용을 서버로 전송하고 'ajax_personal_info.php' 파일을 실행시켜서 서버의 데이터베이스에 업데이트한다. 업데이트를 한 이후에 서버로부터 받은 결과를 alert() 함수를 사용해서 화면에 보여준다.

⑫ 서버에 정상적으로 데이터가 저장되었는지를 확인하기 위해서 '개인정보'를 서버로부터 다시 가져와서 화면에 출력한다. 정상적으로 등록되었다면 불필요한 과정일

수 있지만, 서버의 데이터베이스의 내용이 변경된 것을 확인하는 것은 좋은 습관이다.

[코드 5-17]은 사용자 정보를 보내주는 PHP 코드인데, 로그인을 하게 되면 세션에 사용자 정보가 저장되어 있기 때문에 세션의 정보를 JSON 형식의 문자열로 만들어서 전송한다. 웹 프로그래밍에서 데이터를 JSON 형식으로 만드는 것은 아주 중요한데, 고급 웹 프로그래밍을 하게 되면 인터넷을 통해서 전달되는 많은 데이터를 JSON 형식으로 만들어서 사용한다.

[코드 5-17] 사용자 정보를 보내주는 서버의 PHP 코드 (/data/login/ajax_get_user_info.php)

```
<?php

require_once('../../library/php/session.php');

$ses = new Session();

if(!$ses->is_login()){                                                    //---①
        print 2;
        exit();
}

$data = $ses->get_data_all();                                             //---②

print '{"email":"'.$data["email"].'", "name":"'.$data["name"].'", "cell":"'.$data["cell"].'",
            "addr":"'.$data["addr"].'"}';                                 //---③

exit();

?>
```

① '로그인' 상태가 아니면 브라우저로 2라는 값을 전송해서 '로그인'을 해야 한다는 메시지를 출력한다. 이와 같이 브라우저와 서버 간에 주고받는 값들은 프로그래머가 정하면 되는데, 여기서는 작은 웹사이트이기 때문에 간단한 값을 전달하지만 규모가 큰 사이트에서는 주고받는 값도 시스템 설계자가 규칙을 갖고 모든 값들을 정의한다.

② 세션의 모든 사용자 정보를 가져와서 $data 변수에 넣는다. '로그인'되어서 세션이 이루어졌기 때문에 사용자 정보는 세션이 생성될 때 저장되었다.

③ 사용자 정보를 JSON 형식의 문자열로 만들어서 서버로 전송한다. PHP에서 문자

열을 만드는 것은 작은따옴표(') 또는 큰따옴표(")를 사용하는데, 책의 앞부분에서 설명한 두 개의 따옴표를 사용해서 문자열 안에서 또다른 형태의 문자열을 정의하는 것을 이해하는 것이 중요하다.

[코드 5-18]은 '로그인' 상태에서 진행되는 개인정보를 수정하는 서버의 PHP 코드를 보여준다. 개인정보를 업데이트할 때 빈번하게 진행되는 항목은 6자리의 '임시 비밀번호'로 설정된 것을 사용자의 고유한 비밀번호로 변경하는 항목인데, 이때 이해해야 할 부분은 현재 '임시 비밀번호'가 저장되어 있다면 'auth'(권한)의 설정값이 '0000'으로 되어있기 때문에 정상적인 비밀번호로 변경할때는 'auth'의 값을 '0001'로 변경해야 한다. '지식소프트' 회사는 직원이 1명이기 때문에 'auth'를 사용해서 권한의 범위를 설정하지는 않았지만, 많은 직원이 있는 회사에서는 'auth'의 4자리 숫자를 사용해서 시스템의 접근권한을 정의해야 한다.

[코드 5-18] 개인정보를 수정하는 서버의 PHP 코드 (/data/login/ajax_personal_info.php)

```
<?php

require_once('../../library/php/config.php');
require_once('../../library/php/session.php');
require_once('../../library/php/mysql.php');

$ses = new Session();
$db = new MySQL();

if(!$ses->is_login()){
        print "로그아웃 되었습니다.";
        exit();
}

if(isset($_POST['passwd'])) { $passwd = $_POST['passwd']; }
if(isset($_POST['cell'])) { $cell = $_POST['cell']; }
if(isset($_POST['addr'])) { $addr = $_POST['addr']; }

if(!$db->connect($dbconfig)) {
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

$sql = "SELECT auth FROM User WHERE email='".$ses->get_email()."'";          //---①

$result = $db->query($sql);
if($result === false){
```

```
        $db->close();
        print '{ "error" : "데이터베이스 검색을 실패했습니다." }';
        exit();
}
$data = $db->fetchRow($result);

$db->startTransaction();
if ($data['auth'] == '0000') {                                          //---②
    $sql = "UPDATE User SET passwd=password('".$passwd."'), cell='".$cell."', addr='".$addr."',
                auth='0001' WHERE email='".$ses->get_email()."'";
} else {
    $sql = "UPDATE User SET passwd=password('".$passwd."'), cell='".$cell."', addr='".$addr."'
                WHERE email='".$ses->get_email()."'";
}

$result = $db->query($sql);
if($result == false){
        $db->rollback();
        $db->close();
        print '{ "error" : "데이터베이스 업데이트를 실패했습니다." }';
        exit();
}

$db->commit();

$db->close();

$ses->set_cell($cell);                                                  //---③
$ses->set_addr($addr);

print "개인정보가 수정되었습니다.";
exit();

?>
```

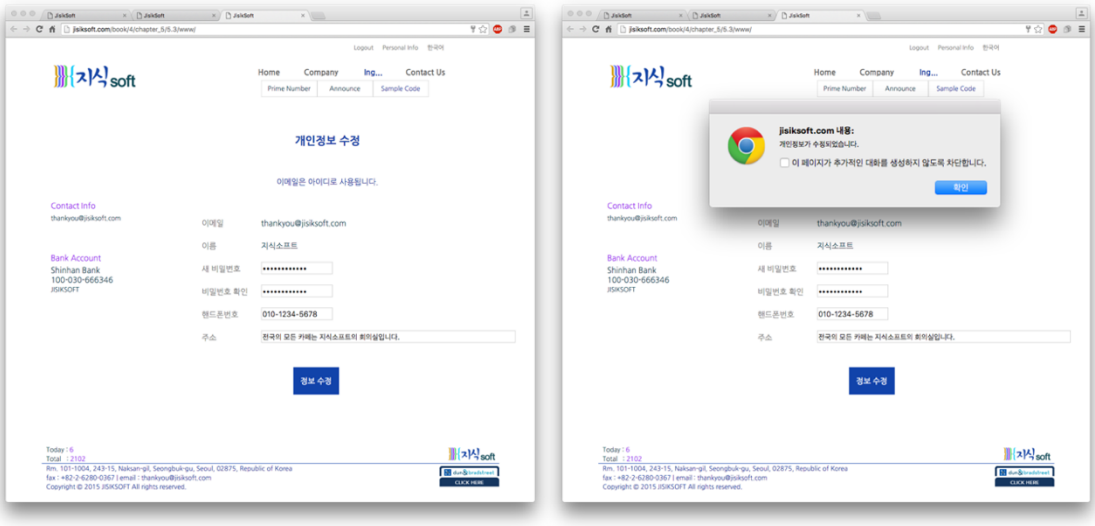
① 세션에 저장된 이메일 주소로 데이터베이스에 등록된 사용자의 권한을 가져온다. Query의 내용은 User 테이블에서(FROM User) 세션의 이메일 주소와 같은 사용자의(WHERE email='".$ses->get_email()."') 권한을 검색해서 가져온다.(SELECT auth)

② 'auth'(권한)의 값이 '0000'이면 '임시 비밀번호'가 저장된 것이기 때문에 'auth'의 값도 '0001'로 변경해야 하는 Query를 만들어야 하며, 그렇지 않은 경우에는 일반 사용자의 정보를 변경하면 된다. Query의 내용은 User 테이블을 업데이트하는데(UPDATE User) 세션의 이메일 주소와 같은 사용자의(WHERE email='".$ses->get_email()."') 모든 정보를 새로운 값으로 설정한다.(SET passwd=password('".$passwd."'), cell='".$cell."', addr='".$addr."')

③ 사용자의 핸드폰번호와 주소가 변경되었을 수 있기 때문에, 세션에 저장된 사용자의 핸드폰번호와 주소를 재설정한다.

[그림 5-23]은 개인정보 수정을 하는 페이지와 수정 후의 결과를 보여주는 메시지 창을 보여준다. 사용자의 정보를 수정하는 과정은 항상 세션이 연결되어 있는지를 서버에서 체크하는 과정에서 시작된다. 사용자의 로그인이 많이 진행되지 않는 사이트들은 세션의 사용에 대해서 크게 신경쓰지 않고 시스템을 구축해도 되지만, 반대로 많은 사용자가 시스템에 로그인한 상태로 있는 시스템에서는 세션의 갯수를 잘 조정해야 한다. 대형 웹서비스를 운영하면 세션의 갯수를 정하는 것이 상당히 중요한데, 최대 세션값보다 많은 사용자가 접속을 시도하면 늦게 접속을 시도하는 사용자는 세션을 연결할 수 없기 때문에 정상적인 서비스를 이용할 수 없다. 세션이라는 것은 소프트웨어의 개념이지만, 세션의 갯수는 하드웨어의 성능에 의존하기 때문에 많은 사용자가 이용하는 웹서비스는 많은 서버를 운용해야 한다.

그림 5-23 '로그인' 이후에는 등록된 이메일과 이름을 제외한 모든 개인정보를 수정할 수 있다.



현재 우리나라에서 보안이 중요하게 여겨지는 많은 온라인 서비스들은 'Active-X'라는 프로그램을 사용하기 때문에 제약이 많다고 하는데, 'Active-X'는 Microsoft 회사에서 만든 것으로 브라우저에서 Windows 프로그램을 사용하는 것으로 이해하면 된다. 인터넷에서는 위험한 코드들도 많이 숨겨져 있는데, 브라우저를 통해서 받은 소프트웨어를 컴퓨터에 설치하는 것은 위험할 때가 많다. 그러나, Explorer라는 브라우저에서만 동작하는 'Active-X'는 인터넷에서 잘 모르고 받은 프로그램을 자신의 컴퓨터에 설치하게 만드는데, 위험한 사이트에서 만들어진 프로그램을 일반 사용자가 잘

못 받을 가능성이 있기 때문에 가능하면 사용하지 않는 것이 좋다. Explorer 브라우저보다 성능이 더 좋은 브라우저가 많기 때문에 보안시장에서 'Active-X'를 사용하지 않는다면 우리나라도 다양한 브라우저들을 많은 사용자가 이용하게 될 것이다. 외국에서는 'Active-X'를 사용하지 않는 은행서비스를 오래전부터 제공해 왔다. 우리나라에서 'Active-X'를 지금도 많은 곳에서 사용하는 이유는 대부분의 보안 모듈들이 'Active-X' 기반으로 만들어졌기 때문이다. 보안 인증서가 추가된 HTTPS(Secure HTTP)를 사용하면 좋은데 HTTPS를 사용할 때 사용하는 인증방식은 외국에서 만든 것이기 때문에, 우리나라의 보안 모듈보다 안전하다고 단정할 수는 없다. 일반적으로 은행을 기업이라고 생각하지만, 은행 시스템의 중요성을 생각하면 국가의 기간산업망이라고 인지하는 것이 좋을 것 같다. 은행의 전산망이 마비되는 순간 국가의 안위도 위협받을 수 있기 때문에 '보안'은 가장 중요한 웹서비스 중의 하나이다.

5.4 게시판 만들기

많은 홈페이지들은 게시판 서비스를 제공하는데, 게시판을 만들 수 있는 편리한 프로그램들이 예전부터 많이 있었다. 대표적인 것이 '제로보드'이며, 예전부터 웹 프로그래머가 아니어도 쉽게 홈페이지를 제작하는 것이 가능했다. 그래서 현재는 고급 프로그래머가 아니어도 디자이너 한 명만 있으면 홈페이지 제작이 가능한 회사를 운영하는 것도 가능한데, 이 책에서 설명하는 코드를 직접 구현하는 방법이 아니라 홈페이지를 쉽게 만들수 있는 제작 툴을 사용하는 방법이 있기 때문이다. 홈페이지를 개인이 처음부터 모든 것을 만드는 것은 어려울 수 있고 시간도 많이 소요되기 때문에 웹 프로그래밍을 하는 많은 사람들이 홈페이지 제작툴을 사용한다.

지금부터 설명하는 게시판 만들기는 화면에 출력하는 내용을 서버의 데이터베이스에 저장하고 브라우저에서 게시판 형태로 출력하는 과정을 보여주는데 독자는 게시판 만드는 방법을 이해하고 자신만의 디자인으로 새로운 게시판을 만들 수 있기를 바란다.

[그림 5-24]는 '지식소프트' 홈페이지의 게시판 페이지를 보여주는데, 일반적인 테이블 형태에 이벤트를 넣어서 내용을 화면에 보여주게 구성하였다.

확인 사이트: http://jisiksoft.com/book/4/chapter_5/5.4/www/

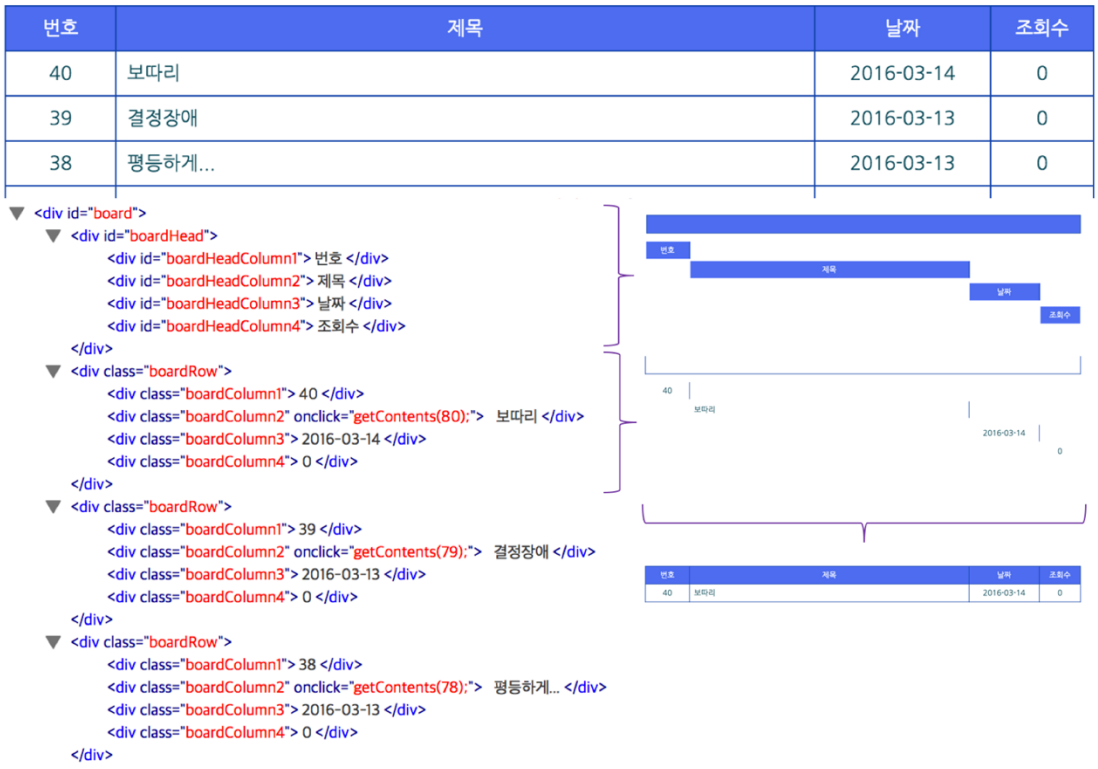
그림 5-24 게시판 페이지

게시판의 제목 리스트를 보여주는 것은 테이블Table 구조로 이루어지는데, 'Chapter 1'의 마지막 부분에서 설명한 <table></table> 태그를 사용해서 만들 수도 있지만 여기서는 태그를 사용해서 게시판을 만드는 방법을 설명하고자 한다. 웹 프로그래밍을 깊이 있게 이해하고 웹 응용프로그램을 만들어본 사람들은 HTML에서 제공하는 많은 태그들을 태그로 대체할 수 있다고 말하는데, 이 책에서 지금까지 사용한 태그들의 90% 이상은 태그였으며 사각형을 만드는 태그만으로 테이블을 만들 수 있다.

[그림 5-25]는 게시판 테이블을 태그만을 사용해서 만드는 방법이다. 사각형의 크기, 바탕색, 테두리 등을 CSS 디자인에서 조정하였으며 게시판 테이블의 내용은 태그 안에 넣어주었다. 또한, 테이블의 제목을 클릭했을 때 해당 항목의 내용을 보여주어야 하기 때문에 'onclick' 이벤트를 사용해서 제목이 클릭되었을 때 JavaScript 코드로 만들어진 getContents() 함수를 실행한다. [그림 5-25]와 같이 게시판의 모든 구조를 개별적으로 만드는 것을 이해하면, 게시판 테이블의

디자인을 웹 프로그래머가 다양하게 변경하는 것이 쉬워진다. [그림 5-25]에서 모든 내용을 감싸는 최상위 태그는 id 값이 'board'인 태그이며 전체 게시판 테이블을 포함하기 때문에 게시판의 위치 조정을 여기서 하면 된다. 게시판 테이블에서 태그의 클래스와 id 값을 사용해서 디자인을 만들어주면 된다. 필자의 경험으로는 CSS 디자인을 변경하는데 긴 시간이 필요했으니 독자도 여유를 가지고 디자인 작업을 진행하기를 바란다.

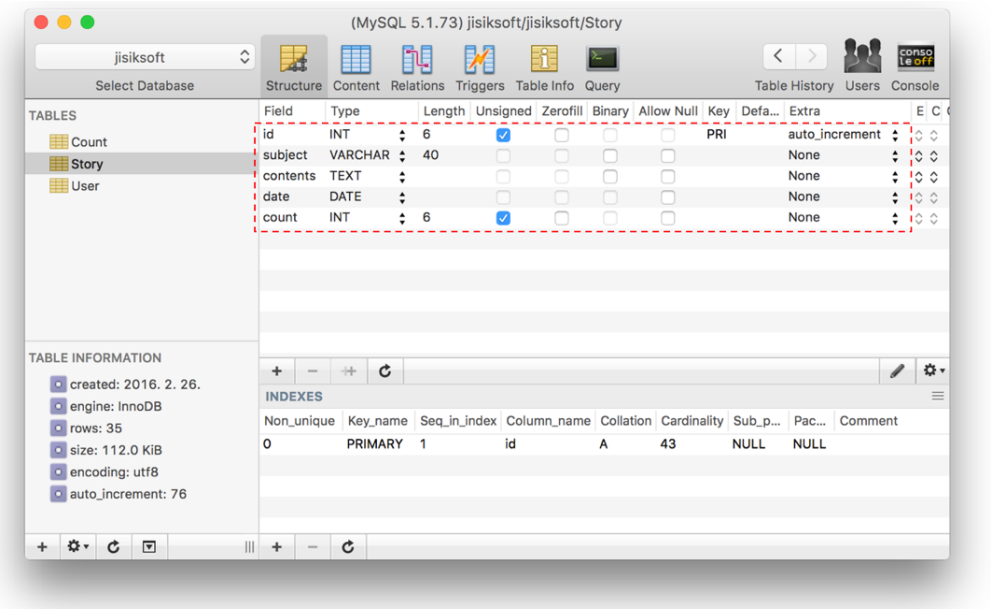
그림 5-25 태그로 만들어진 게시판의 태그 구조



게시판의 내용을 데이터베이스에 저장하기 위해서는 하나의 게시판을 위해서 하나의 데이터베이스 테이블을 만든다. '지식소프트' 홈페이지에서는 게시판이 한 개이기 때문에 데이터베이스에서 'Story'라는 테이블 한 개를 구축했다. 게시판을 위해서 다양한 정보를 데이터베이스에 넣을 수 있는데, 여기서 만드는 게시판은 id, 제목Subject, 내용Contents, 날짜Date, 조회수Count 정보만을 저장한다. '지식소프트' 홈페이지는 1명의 사용자만 게시판에 글을 작성하기 때문에 게시판에 글을 쓴 사용자가 누구인지 저장할 필요가 없는데, 실제 데이터베이스에는 글을 작성한 사람의 이메일 주소(이 책에서는 email이 id로 사용된다.)도 같이 저장해야 한다. 게시판의 각각의 내용을 구분하기 위해서 사용되는 id는 정수값으로 내용이 추가될 때마다 자동으로 1씩 증가한 id 값을 갖게 되는데, 이렇게 자동으로 증가하는 값의 설정은 데이터베이스에서 제공하고 있다.

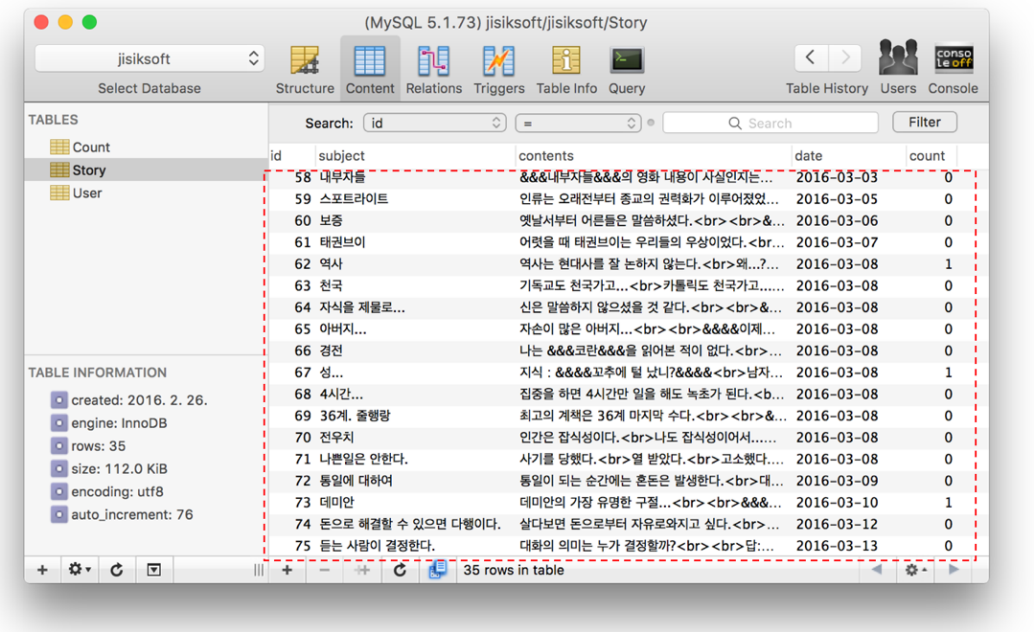
[그림 5-26]은 Story 테이블의 구조를 보여주는데, 'id' 값은 Primary Key로 설정되었고 'auto_increment'로 설정해서 자동으로 1씩 증가한다. 이와 같이 데이터베이스에서는 id값을 자동으로 증가하는 정수를 사용할 때가 많은데 데이터베이스에 데이터를 저장할 때 특별히 id 값의 중복을 확인할 필요가 없기 때문에 편리하다. 데이터베이스에 저장되는 내용의 크기도 테이블 구조를 만들 때 정확히 선언해 주어야 하는데, 제목은 40-bytes의 문자열까지 저장할 수 있으며 게시판의 내용은 TEXT 데이터형Data Type으로 선언해서 문자열의 크기를 정하지 않았다. 데이터베이스에서는 저장되는 데이터의 크기가 정해지면 데이터에 접근하는 속도가 빠르고 이와 같이 TEXT로 설정하면 상대적으로 내용을 가져오는게 느린 편인데, 컴퓨터는 워낙 빠르기 때문에 데이터 형에 따른 속도의 차이를 사람들이 인지할 수 없다. 그래서 많은 사용자가 접속하는 사이트가 아니라면 데이터베이스를 설계할 때 부담은 많이 줄어든다. 인터넷 사이트에서 입력하는 정보의 텍스트 크기가 정해진 곳이 의외로 많은데, 데이터베이스에 큰 무리를 주지 않기 위해서 데이터 형을 TEXT로 설정하지 않았다고 이해하면 된다.

그림 5-26 게시판 내용을 저장하기 위해 만들어진 Story 테이블



[그림 5-27]은 Story 테이블에 다수의 내용이 입력된 결과를 보여주는데, 데이터베이스에는 몇만 개의 데이터가 저장되어도 상당히 빠르게 정보를 검색할 수 있다.

그림 5-27 Story 테이블에 게시판 내용이 저장된 결과

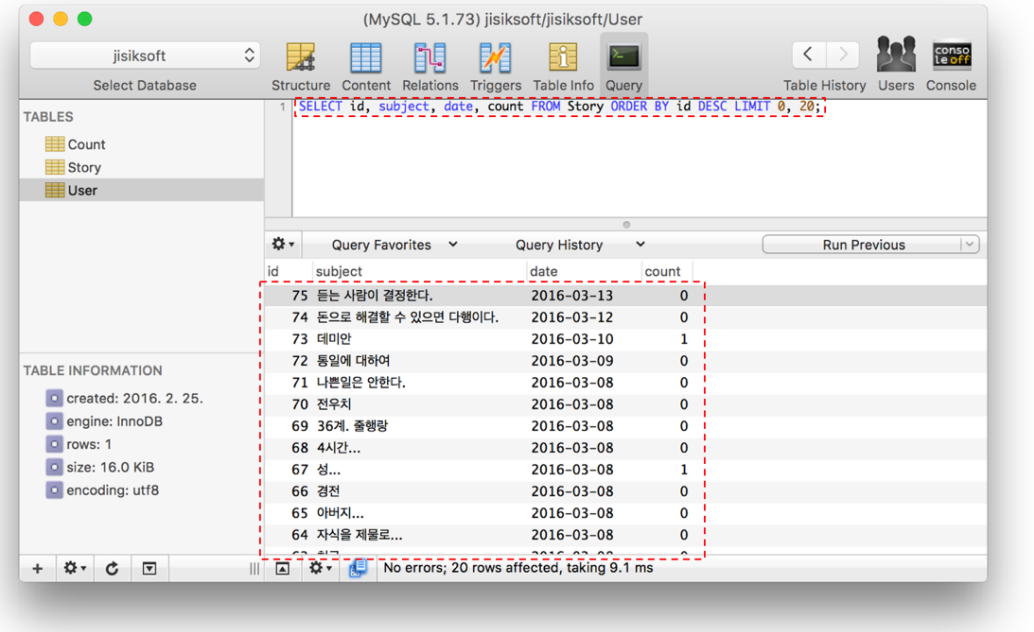


[그림 5-28]은 게시판 테이블에 20개의 정보들을 보여주기 위해서 데이터베이스로부터 20개의 내용만을 검색해서 가져오는 Query의 실행 결과를 보여주는데, Query의 명령은 아래와 같다.

SELECT id, subject, date, count FROM Story ORDER BY id DESC LIMIT 0, 20;

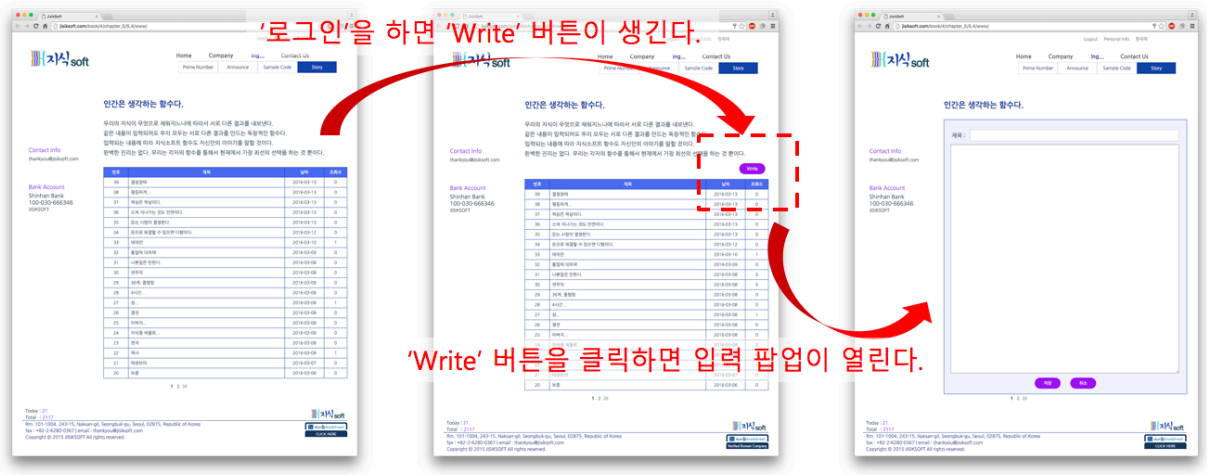
Query의 내용은 Story 테이블에서(FROM Story) id를 내림차순으로 정렬하고(ORDER BY id DESC) 위에서 0번째부터 시작해서 20개 데이터들의(LIMIT 0, 20) id, 제목, 날짜, 조회 수를 검색해서 가져온다.(SELECT id, subject, date, count)

그림 5-28 최근에 저장된 20 개의 데이터를 조회한 Query 결과



게시판 페이지는 로그인을 해야 글을 작성할 수 있는데, [그림 5-29]는 '로그인'을 하면 글을 작성하기 위한 'Write' 버튼이 화면에 보여지는 그림이다. 'Write' 버튼을 클릭하면 글을 작성할 수 있는 에디터 창이 보여지며, 게시판의 내용을 작성해서 저장하는 편리한 환경을 만들었다. 인터넷에서 글을 작성하는 환경을 제공하는 많은 곳에서 게시판 서비스를 제공하기 때문에 독자들도 게시판의 사용방법을 쉽게 이해할 수 있으며, '로그인' 상태에서 글을 작성할 수 있다는 것은 세션이 연결되었는지를 확인한 이후에 'Write' 버튼을 화면에서 보여지게 만들면 된다. 글을 작성한 후에 서버에 저장할 때에도 서버에서는 정상적인 세션이 이루어졌는지를 확인하고 데이터베이스에 저장한다.

그림 5-29 '로그인' 이후에 게시판 내용을 작성하기 위한 'Write' 버튼이 생성된 결과

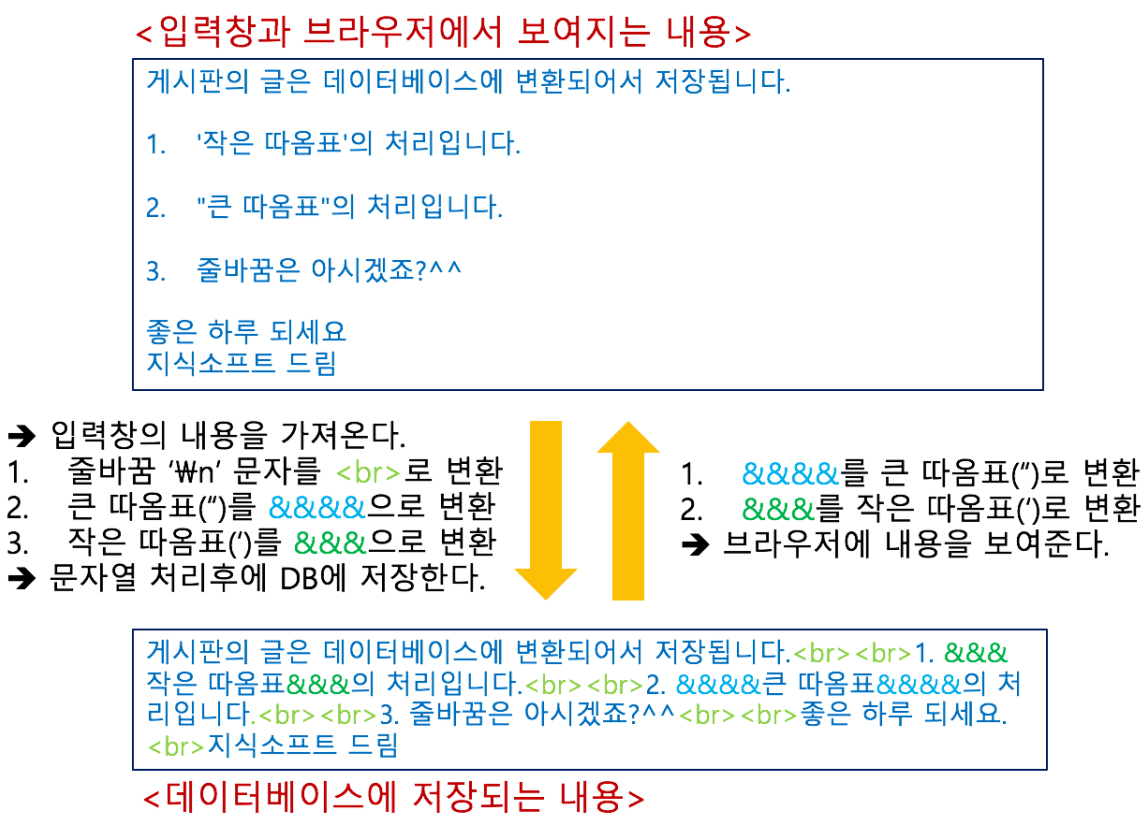


게시판의 글을 작성할 때는 입력할 수 있는 <input> 태그와 장문의 글을 입력하는 <textarea> 태그를 사용하는데, 이러한 태그들에 입력되는 글자는 일반적인 프로그래밍의 데이터 형식을 따른다. 줄바꿈 문자는 '₩n'의 값을 가지고 있는데, 게시판에서 내용을 확인할 때 브라우저에서는 줄바꿈을 해주기 위해서는
 태그를 사용해야 한다. 그래서, 서버에 게시판의 글을 저장하기 전에는 '₩n' 문자를
 태그로 변경해서 저장하는 것이 좋다. 또 한가지 생각해야 할 부분은 서버에서 Query를 사용해서 데이터를 저장하는데 Query 명령어를 만들 때 PHP 언어의 문자열을 만드는 두 개의 따옴표들(', ")을 사용하므로, 게시판의 글의 내용에 따옴표(', ")가 포함되어 있으면 서버에서 데이터베이스를 사용하는데 문제가 발생한다. 그래서, 서버에 게시판의 글을 전달하기 전에 따옴표(', ")들을 특정 문자로 변경 해주는 것이 좋다. 필자는 두 개의 따옴표들을 잘 사용하지 않는 문자열로

변환하는 과정을 만들었는데, 큰 따옴표(")는 '&&&&' 문자열로 변환을 하고, 작은 따옴표(')는 '&&&' 문자열로 변환해주었다. 이렇게 특정 문자열로 변경하는 것은 프로그래머가 임의로 정하면 되는데, 다수의 '&' 문자를 여러 번 반복해서 사용하는 경우가 거의 없기 때문에 필자는 '&' 문자들을 가지고 변환과정을 진행하였다. 물론 지금 만드는 게시판은 필자만이 글을 쓰기 때문에 이와 같이 정한 것인데, 만약 다수의 사용자가 글을 작성한다면 '&' 문자를 세 번 이상 중복해서 사용하는 것을 금지시키는 조건을 만들어 줄 필요가 있다.

[그림 5-30]은 브라우저에서의 입력 내용을 데이터베이스에 저장하기 전에 문자열을 변경하는 과정과 데이터베이스로부터 게시판 내용을 받아와서 따옴표들로 변환한 후 브라우저에 출력하는 과정을 보여준다. 서버에서 데이터베이스와의 통신을 위해서 만들어지는 Query 명령어가 문자열로 만들어지기 때문에 이와 같이 따옴표를 특수문자열로 변환하는 과정을 항상 진행해야 한다. 문자열을 변환하는 과정은 서버에서 진행해도 되지만, 웹 프로그래밍을 할 때 가장 염두에 두어야 할 것은 서버에서 해야할 일을 최소화시켜야 한다는 것이다. 서버의 기본은 다수의 사용자에게 서비스를 제공하기 때문에 서버에서 동작하는 프로그램은 간단할수록 좋다. 그래서, [그림 5-30]에서와 같이 항상 진행되는 문자열 변환 과정을 이 책에서는 사용자 브라우저에서 수행하도록 JavaScript 언어에서 구현했다.

그림 5-30 입력된 문자열을 특수문자로 변환 후 데이터베이스에 저장하는 방법



'지식소프트' 홈페이지에는 게시판 페이지가 하나뿐이기 때문에 게시판을 구현하는 'smenu_3_4.html' 파일에서 HTML, JavaScript, CSS 코드를 모두 구현하였다. 하지만 하나의 페이지에서만 사용하는 JavaScript 함수들과 CSS 디자인 설정들을 홈페이지 전체에서 사용하는 jisiksoft.js 파일과 jisiksoft.css 파일에 넣는 것은 좋지 않다.

[코드 5-19]는 게시판 페이지의 HTML 코드를 보여주며, [코드 5-20]은 게시판의 이벤트를 수행하는 JavaScript 코드를 보여준다. 실제 모든 코드는 하나의 파일에 저장되었으며, CSS 디자인 설정들은 설명을 생략한다.

[코드 5-19] 게시판 페이지의 HTML 코드 (/data/container/smenu_3_4.html)

```
<br><br>
<div class="big bold color1">인간은 생각하는 함수다.</div>
<br>
우리의 지식이 무엇으로 채워지느냐에 따라서 서로 다른 결과를 내보낸다.<br>
같은 내용이 입력되어도 우리 모두는 서로 다른 결과를 만드는 독창적인 함수다.<br>
입력되는 내용에 따라 지식소프트 함수도 자신만의 이야기를 말할 것이다.<br>
완벽한 진리는 없다. 우리는 각자의 함수를 통해서 현재에서 가장 최선의 선택을 하는 것뿐이다.<br>
<br>
<div class="button buttonArea" id="boardButtonWrite" onclick="showPopupWrite();">
        Write</div>
                                                                    //---1
<div id="board"></div>                                              //---2

<div id="boardPopupRead">                                           //---3
    <div class="button" id="hideButton" onclick="hideText();">X</div>   //---4
    <div id="boardText">
        <div class="big bold color1" id="textTitle"></div>          //---5
        <br><br>
        <div id="textContents"></div>
    </div>
    <div class="button" id="hideButton" onclick="hideText();">X</div>
    <div class="buttonArea" id="boardReadButtonArea">               //---6
        <div class="button btnBoard" id="firstButton" onclick="editArticle();">수정</div>
        <div class="button btnBoard" id="secondButton" onclick="showAlertToDelete();">
            삭제</div>
    </div>
    <br>
</div>

<div id="boardPopupWrite">                                          //---7
```

```
    <br>     제목 :
    <input type="text" id="boardWriteTitle"></input>                    //---8
    <br>
    <textarea id="boardWriteContents" rows="35" cols="76"></textarea>   //---9
    <br>
    <div class="buttonArea" id="boardWriteButtonArea">                  //---10
        <div class="button btnBoard" id="firstButton" onclick="saveArticle();">저장</div>
        <div class="button btnBoard" id="secondButton" onclick="updateArticle();">수정</div>
        <div class="button btnBoard" id="thirdButton" onclick="cancelArticle();">취소</div>
    </div>
    <br>
</div>

<div id="boardPopupAlert">                                              //---11
    <div style="height:30px;"></div>
    <div id="textAlert" style="position:relative;left:50px;">정말 삭제하시겠습니까?</div>
    <div style="height:30px;"></div>
    <div id="boardAlertButtonArea">
        <div class="button" id="firstButton" onclick="deleteArticle();">예</div>
        <div class="button" id="secondButton" onclick="hideAlert();">아니오</div>
    </div>
    <br>
</div>
```

① '로그인'을 한 이후에 보여지는 'Write' 버튼을 태그를 사용해서 만들었으며, 클릭하면 showPopupWrite() 함수를 호출해서 게시판의 글을 쓸 수 있는 창이 열린다.

② 게시판의 리스트를 테이블 형태로 보여주기 위한 'board'라는 id 값을 가진 객체로서, 테이블의 내용은 서버로부터 받아온 값에 따라 변경되기 때문에 객체에 들어갈 내용을 JavaScript 함수에서 만들어서 넣어준다.

③ 게시판 테이블에 나열되는 제목들 중에 하나를 클릭하면 내용을 보여주는데, 화면에 보이지 않던 'boardPopupRead'라는 id 값을 가진 객체에 해당 게시물의 내용을 넣어서 화면에 보여준다. 게시판에서 내용을 보여주는 객체들은 이와 같이 HTML 코드로 미리 만들어져서 화면에 숨겨져 있다.

④ 게시물의 내용을 화면에 보여줄 때, 오른쪽에는 'X' 버튼이 존재하며, 'X' 버튼을 클릭하면 게시물의 내용이 화면에서 사라지게 된다. 이 책에서 만든 모든 버튼은 이와 같이 태그를 사용해서 만들었다.

⑤ 게시판의 내용을 보여줄 때 게시글의 제목은 'textTitle'이라는 id 값의 태그에 넣고, 내용은 'textContents'라는 id 값의 태그에 넣는다.

⑥ '로그인'을 한 사용자는 게시글을 '수정'이나 '삭제'를 할 수 있는데, '로그인'을 한 사용자라면 '수정'과 '삭제' 버튼을 화면에서 보게 된다.

⑦ 게시판에 글을 작성할 때 보여지는 입력 창은 'boardPopupWrite'라는 id 값을 가진 태그 안에 만들어져 있으며, '로그인' 상태에서 볼 수 있는 'Write' 버튼을 클릭하면 화면에 보여지게 된다.

⑧ 제목은 'boardWriteTitle'이라는 id 값의 <input> 태그에서 입력받는다.

⑨ 작성하는 글의 내용은 'boardWriteContents'라는 id 값의 <textarea> 태그에서 입력받는다.

⑩ 글을 새로 작성하거나 수정할 때 보여지는 버튼은 'boardWriteButtonArea'라는 id 값을 가진 태그 안에 있으며, 새로운 글을 작성할 때는 '저장' 버튼이 입력 창에 보여지고 글을 수정할 때는 '수정' 버튼이 입력 창 밑에 보여진다.

⑪ 글을 취소할 때는 "정말 삭제하시겠습니까?"라는 메시지 창을 보여주고 삭제하기 전에 한번 더 사용자에게 물어보는 과정이 필요하다. 'boardPopupAlert'는 삭제하기 전에 한번 더 물어보는 메시지 창의 내용을 가지고 있으며, '예' 버튼을 클릭하면 deleteArticle() 함수를 실행해서 해당 게시물을 서버의 데이터베이스에서 지우는 과정을 진행한다.

[코드 5-20] 게시판 페이지의 이벤트 처리를 위한 JavaScript (/data/container/smenu_3_4.html)

```
<script>

    var countInPage = 20;                                    //---①
    var currArticle = 0;                                     //---②
    var currPage = 0;                                        //---③

    function setQuotationMark(input) {                       //---④

        var output;

        var arrString = input.split('&&&&');
        output = '';
        for (var i=0; i<arrString.length; i++) {
                output += arrString[i];
                if (i != (arrString.length - 1))
                        output += "'";
        }
        arrString = output.split('&&&');
        output = '';
```

```
    for (var i=0; i<arrString.length; i++) {
            output += arrString[i];
            if (i != (arrString.length - 1))
                    output += """;
    }

    return output;
}

function unsetQuotationMark(input) {                                    //---⑤

    var output;

    var arrString = input.split("");
    output = '';
    for (var i=0; i<arrString.length; i++) {
            output += arrString[i];
            if (i != (arrString.length - 1))
                    output += '&&&&';
    }
    arrString = output.split("""");
    output = '';
    for (var i=0; i<arrString.length; i++) {
            output += arrString[i];
            if (i != (arrString.length - 1))
                    output += '&&&';
    }

    return output;
}

function makeBoard(noPage) {                                            //---⑥

    var param = { board: 'Story', page: noPage };
    var jsonData = doAjax('./data/board/ajax_get_board.php', param);

    var obj = JSON.parse(jsonData);

    currPage = obj.page;                                                //---⑦

    var no = obj.count - (obj.page * countInPage);                      //---⑧

    var str = '<div id="boardHead">';                                   //---⑨
    str += '<div id="boardHeadColumn1">번호</div> \
                    <div id="boardHeadColumn2">제목</div> \
                    <div id="boardHeadColumn3">날짜</div> \
                    <div id="boardHeadColumn4">조회수</div>';
```

```
    str += '</div>';

    for (var i=0; i<obj.boardPage.length; i++) {                    //---⑩
            str += '<div class="boardRow">';
            str += '<div class="boardColumn1">'+no+'</div> \
              <div class="boardColumn2" onclick="showArticle('+obj.boardPage[i].id+');"> \
                '+setQuotationMark(obj.boardPage[i].subject)+'</div> \
              <div class="boardColumn3">'+obj.boardPage[i].date+'</div> \
              <div class="boardColumn4">'+obj.boardPage[i].count+'</div>';
            str += '</div>';

            no -= 1;
    }

    str += '<div style="position:relative;top:10px;left:200px;width:300px;height:30px;">';
                                                                    //---⑪
    if (obj.page > 0) {
        str += '<div class="button2" onclick="makeBoard('+(obj.page - 1)+');"> << </div>';
    }
    var startIndex = obj.page - 4;                                  //---⑫
    if (startIndex < 0) {
            startIndex = 0;
    }
    var lastIndex = Math.ceil(obj.count / countInPage);
    var endIndex = obj.page + 5;
    if (endIndex > lastIndex) {
            endIndex = lastIndex;
    }

    for (var i=startIndex ; i<endIndex; i++) {                      //---⑬
            if (i == obj.page) {
                str += '<div class="bold black" style="width:20px;float:left;">'+ (i+1) +'</div>';
            }
            else {
                str += '<div class="button2" onclick="makeBoard('+i+');">'+ (i+1) +'</div>';
            }
    }
    if (no > 0) {
        str += '<div class="button2" onclick="makeBoard('+(obj.page + 1)+');"> >> </div>';
    }
    str += '</div><br><br>';

    $('#board').empty().html(str);                                  //---⑭
}

function showArticle(idArticle) {                                   //---⑮

    var param = { board: 'Story', id: idArticle };
```

```
    var jsonData = doAjax('./data/board/ajax_get_article.php', param);

    var obj = JSON.parse(jsonData);

    var text = setQuotationMark(obj.article.subject);
    $('#textTitle').empty().html(text);

    text = setQuotationMark(obj.article.contents);
    $('#textContents').empty().html(text);

    $('#boardPopupRead').css({"display":"block"});

    currArticle = idArticle;
}

function hidePopupRead() {                                          //---⑯
    $('#boardPopupRead').css({"display":"none"});
}

function showPopupWrite() {                                         //---⑰
    $('#boardWriteTitle').val('');
    $('#boardWriteContents').val('');
    $('#boardPopupWrite #firstButton').css({"display":"block"});    //---⑱
    $('#boardPopupWrite #secondButton').css({"display":"none"});
    $('#boardPopupWrite').css({"display":"block"});
}

function saveArticle() {                                            //---⑲

    var text = $('#boardWriteTitle').val();
    var textTitle = unsetQuotationMark(text);

    text = $('#boardWriteContents').val();
    var arrString = text.split('\n');
    text = '';
    for (var i=0; i<arrString.length; i++) {
            text += arrString[i];
            if (i != (arrString.length - 1))
                    text += '<br>';
    }
    var textContents = unsetQuotationMark(text);

    var param = { board: 'Story', title: textTitle, contents: textContents };
    var result = doAjax('./data/board/ajax_save_article.php', param);

    if (result == 0) {                                              //---⑳
            $('#boardWriteTitle').val('');
            $('#boardWriteContents').val('');
            $('#boardPopupWrite').css({"display":"none"});
```

```
                makeBoard(0);
    }
}

function cancelArticle() {                                          //---㉑
    $('#boardPopupWrite').css({"display":"none"});
}

function editArticle() {                                            //---㉒

    var title = $('#textTitle').html();
    var text = $('#textContents').html();

    var arrString = text.split('<br>');
    text = '';
    for (var i=0; i<arrString.length; i++) {
                text += arrString[i];
                if (i != (arrString.length - 1))
                        text += '\n';
    }

    $('#boardWriteTitle').val(title);
    $('#boardWriteContents').val(text);
    $('#textTitle').html('');
    $('#textContents').html('');

    $('#boardPopupRead').css({"display":"none"});

    $('#boardWriteButtonArea #firstButton').css({"display":"none"});
    $('#boardWriteButtonArea #secondButton').css({"display":"block"});
    $('#boardPopupWrite').css({"display":"block"});
}

function updateArticle() {                                          //---㉓

    var text = $('#boardWriteTitle').val();
    var textTitle = unsetQuotationMark(text);

    text = $('#boardWriteContents').val();
    var arrString = text.split('\n');
    text = '';
    for (var i=0; i<arrString.length; i++) {
                text += arrString[i];
                if (i != (arrString.length - 1))
                        text += '<br>';
    }
    var textContents = unsetQuotationMark(text);

    var param = { board: 'Story', id: currArticle, title: textTitle, contents: textContents };
```

```
    var result = doAjax('./data/board/ajax_update_article.php', param);

    if (result == 0) {
            $('#boardPopupWrite').css({"display":"none"});

            makeBoard(currPage);
    }
}

function showAlertToDelete() {                                          //---㉔
    $('#boardPopupAlert').css({
                            "display":"block",
                            "left":($(window).width() / 2) - 30,
                            "top":($(window).height() / 2) - 100
                            });
}

function deleteArticle() {                                              //---㉕
    $('#boardPopupAlert').css({"display":"none"});

    var param = { board: 'Story', id: currArticle };
    var result = doAjax('./data/board/ajax_delete_article.php', param);

    if (result == 0) {
            $('#boardPopupRead').css({"display":"none"});

            makeBoard(currPage);
    }
}

function hidePopupAlert() {                                             //---㉖
    $('#boardPopupAlert').css({"display":"none"});
}

makeBoard(0);                                                           //---㉗

if (isLogedin() == 1) {                                                 //---㉘
    $('.buttonArea').css({"display":"block"});
    $(window).unbind("mousedown");
} else {
    $('.buttonArea').css({"display":"none"});
    $(window).bind("mousedown", function(){ return false; });
}

</script>
```

① 게시판 테이블에 보여지는 리스트의 갯수는 countInPage 변수에 저장되어 있으며, 20으로 설정되어 있기 때문에 최대로 보여지는 갯수는 20개이다. 이와 같이 전

역변수에 값을 설정해서 프로그래밍을 하는 것은 모든 프로그래밍에서 기본으로 사용되는 방법이다.

② 서버의 데이터베이스에 작성된 글이 저장될 때 자동으로 부여된 id 값을 저장하기 위해서 사용되는 변수로서, 현재Current 하나의 게시물Article의 내용을 화면에서 보고 있다면 currArticle에 id 값이 저장되어 있으며 게시물의 내용을 수정하거나 삭제할 때 idArticle에 저장된 값을 사용해서 서버의 데이터베이스에 저장된 항목을 수정하거나 삭제한다.

③ 게시판에 저장된 내용이 많다면 20개의 내용만 한 화면에 보여주게 되는데, 데이터베이스에 저장된 많은 게시물을 20개 단위로 나누어서 페이지로 보여준다. 화면에 보여지는 게시판 리스트가 몇 번째 페이지의 내용인지를 알기 위해서 현재Current 페이지Page의 값을 currPage 변수에 저장한다. 데이터베이스에 저장된 내용의 갯수와 현재 화면에 보여지는 페이지의 값을 알 수 있기 때문에 게시판의 페이지가 변경될 때마다 가져와야 하는 항목들을 제어할 수 있다.

④ 서버로부터 받은 게시물의 제목과 내용에는 따옴표(', ")를 '&&&'와 '&&&&'의 특수문자들로 변환되어 있기 때문에, 화면에서 정상적으로 따옴표들을 표시하기 위해서 문자열 변환작업을 진행해야 한다.([그림 5-30] 참조) 'setQuotation()' 함수는 서버로부터 받은 데이터의 특수문자들을 따옴표로 변환한다.

⑤ 게시판에 글을 작성하거나 수정할 때에는 서버의 데이터베이스에 작성된 내용을 저장하는데, 저장하기 전에 브라우저에서 따옴표(', ")를 '&&&'와 '&&&&'의 특수문자들로 변경해야 한다.([그림 5-30] 참조) 서버의 PHP 언어에서 데이터베이스의 Query 명령어를 사용할 때, 따옴표를 사용하기 때문에 데이터베이스에 입력되는 내용에는 따옴표가 존재하면 안된다. 게시판의 글의 내용이 추가되거나 수정될 때에는 서버로 보내기 전에 항상 unsetQuotationMark() 함수를 사용해서 따옴표를 특수문자로 변경하는 과정을 진행한다.

⑥ 서버로부터 게시판 테이블의 내용을 받아서 화면에 보여주는 함수이며, 서버로부터 가져올 페이지는 매개변수인 noPage에서 정해진다. 서버로부터 해당 페이지의 내용을 요청하고 받은 후에 게시판을 만드는데, 게시판의 모든 내용이 태그들로 이루어진 객체인 것을 이해해야 한다. 객체에서는 이벤트를 발생시킬 수가 있는데, 게시판에서 제목을 보여주는 태그에 onclick 이벤트를 만들어서 클릭했을 때 해당 게시물을 화면에 보여준다. 함수 안에서 주의 깊게 봐야할 것은 다수의 객체들을 문자열로 만들어서 객체 안에 추가하는 것이며, 게이판 테이블

을 태그를 사용해서 만든 것이다.

⑦ 서버로부터 받은 데이터는 JSON 형식이며, page 항목의 값은 현재 가져온 페이지의 숫자를 가지고 있기 때문에 현재Current 페이지Page의 값을 저장하는 전역변수 currPage에 값을 넣는다. 현재 페이지의 정보를 저장하는 이유는 게시판의 글이 수정된 이후에 현재 페이지를 다시 한번 화면에 출력해서 수정에 의한 변경된 내용을 화면에서 보여주기 위해서다.

⑧ 서버로부터 받은 정보에서 count 항목의 값은 전체 게시물의 총 갯수의 값을 가지고 있는데, 현재 페이지의 값과 화면에서 보여주는 갯수(20개)를 곱한 값을 빼면 현재 페이지에서 출력되는 항목들의 시작하는 번호를 알 수 있다. 즉, no 변수는 게시판의 왼쪽에 보여지는 숫자의 가장 위에서 시작하는 값을 갖는다.

⑨ 게시판의 태그들을 문자열로 만들어서 str 변수에 정장한다. 게시판에 나타나는 내용은 페이지의 내용에 따라서 변경되기 때문에 JavaScript를 사용해서 객체를 생성하는 과정이 항상 진행된다.

⑩ 서버로부터 받은 JSON 데이터에서 boardPage 항목은 화면에 출력되는 게시물들의 정보들을 배열로 가지고 있으며, for loop을 사용해서 모든 게시물들의 내용들을 태그 객체에 넣어준다. 제목을 클릭하면 게시물의 내용을 보여주기 위해서 onclick 이벤트가 발생했을 때 showArticle() 함수를 실행하는데 매개변수로 해당 게시물의 id를 전달한다. 또한 제목의 내용을 게시판에서 보여주기 위해서는 setQuotationMark() 함수를 사용해서 따옴표(', ")를 화면에서 출력할 수 있게 만들었다. 또한, for loop의 아래에서 게시물의 번호를 의미하는 no 변수를 1씩 빼서 출력하는 번호를 차례대로 감소한다.

⑪ 게시판의 아래에 페이지의 숫자들이 나오는데, 현재 페이지의 값을 가지고 페이지를 이동할 수 있는 숫자들의 객체를 만든다.

⑫ 페이지를 이동하는 숫자를 계산하기 위해서 현재 페이지가 중간 페이지라면 좌측에 보여지는 숫자의 최소값은 0이거나 현재 페이지에서 4를 뺀 숫자 이상이어야 한다. 마찬가지로 현재 페이지 숫자의 오른쪽에는 최대값이 현재 페이지에 5를 더한 숫자보다 작아야 한다. 게시판의 아래에 페이지 이동을 위한 숫자의 최소값은 startIndex 변수에 저장하고, 최대값은 endIndex 변수에 저장한다.

⑬ 게시판의 아래에 페이지를 이동하는 숫자들을 for loop을 사용해서 만드는데, 현재 페이지의 숫자이면 굵은 색으로 보이고 이동할 수 있는 페이지들은 onclick 이벤트를 사용해서 makeBoard() 함수를 실행해서 새로운 페이지를 화면에 보여준다.

⑭ 게시판의 모든 객체들의 내용이 문자열로 만들어지면 'board'라는 id 값을 가진 태그 객체에 넣어서 화면에 출력한다.

⑮ 게시판의 내용을 화면에 보여주는 함수이며, 게시판의 내용을 저장하고 있는 서버의 데이터베이스 테이블 이름(Story)과 게시물의 id를 서버로 전송하고 'ajax_get_article.php' 함수를 실행시켜서 게시물의 정보을 JSON 형식으로 받아온다. 받아온 JSON 데이터에는 게시물의 제목과 내용이 있으며, setQuotationMark() 함수를 사용해서 따옴표(', ")의 변환을 실행한 후 HTML 객체에 넣어준다. 게시물의 정보는 'boardPopupRead'라는 id 값을 가진 태그 객체에 있기 때문에 해당 객체를 'display' 속성을 사용해서 화면에서 볼 수 있게 만든다. 현재 화면에서 보여지는 게시물의 id를 currArticla에 저장하는데, 이후에 게시물을 수정거나 삭제할 때 사용한다.

⑯ 게시물을 읽다가 오른쪽의 'X' 버튼을 클릭하면 'display' 속성을 'none'으로 설정해서 화면에서 사라지게 한다.

⑰ 로그인한 상태에서 게시판 화면에는 글을 작성할 수 있는 'Write' 버튼이 보이는데, 'Write' 버튼이 클릭되면 showPopupWrite() 함수가 실행되어 글을 작성할 수 있는 창이 화면에 보이게 된다. 글을 새로 작성하는 것이기 때문에 제목과 내용의 입력 창을 빈 공간으로 만들어주고, '저장' 버튼이 화면에 보이고 '수정' 버튼은 화면에서 보이지 않게 디자인을 설정했다.

⑱ CSS와 jQuery에서 객체에 접근하기 위해서는 클래스나 id 값을 사용하는데, 다수의 id 값들을 사용해서 특정 객체에 접근하는 것이 가능하다. $('#boardPopupWrite #firstButton')와 같이 사용한 것은 'boardPopupWrite'라는 id를 가진 객체에 접근하고 해당 객체 안에 있는 'firstButton'이라는 id 값을 가진 객체에 다시 접근해서 가져오게 된다. 이와 같이 사용하면, 'firstButton'이라는 값을 중복해서 사용해도 특정 위치의 객체에 접근할 수 있다.

⑲ 새로운 글을 작성해서 서버에 저장하기 위해서 실행되는 함수로서 [그림 5-30]에서 보여지는 과정을 진행하고 서버에 저장한다. 즉, 입력 창으로부터 게시물의 제목과 내용을 가져와서 unsetQuotationMark() 함수를 사용해서 따옴표(', ")들을 특수문자열('&&&', '&&&&')들로 변경하고 서버에 저장한다. 더하여 브라우저에서는 줄바꿈 태그가
이고 입력 창에서는 줄바꿈 문자가 '₩n'이기 때문에 '₩n' 문자를
 태그로 변경하는 과정이 추가되었다.

⑳ 새로 작성한 글이 정상적으로 서버에 저장되면 0을 결과값으로 받으며, 이 때

작성을 위한 입력 창을 화면에서 사라지게 하고 makeBoard() 함수를 사용해서 새로운 항목이 추가된 게시판을 화면에 출력한다.

㉑ 게시판에 내용을 넣기 위하여 생성된 입력창에서 'X' 버튼이 클릭되면 cancelArticle() 함수를 실행시켜서 입력 창을 화면에서 사라지게 한다.

㉒ '로그인' 상태에서는 게시물의 내용을 수정할 수 있는데, '수정' 버튼이 클릭되면 글의 수정이 가능한 입력 창이 생성된다. 수정을 위한 입력 창은 새로 글을 작성하는 입력 창과 같지만, 제목과 내용에는 수정하기 위한 게시물의 정보들이 삽입되었다. 'firstButton'이라는 id를 가진 태그는 '저장' 버튼이기 때문에 입력 창에서 숨겨지고, 'secondButton'이라는 id를 가진 태그는 '수정' 버튼이기 때문에 입력 창에서 보이는 CSS 디자인 작업을 추가했다.

㉓ 글을 수정한 이후에 '수정' 버튼이 클릭되면 updateArticle() 함수가 실행되는데, 새로운 글을 등록하는 saveArticle() 함수와 내용이 거의 같고 다른 부분은 저장되었던 현재 게시물의 데이터베이스에서 사용하는 id 값(currArticle)을 포함한 정보를 서버에 전송한다. 서버의 데이터베이스에 정상적으로 업데이트가 완료되면 결과값으로 0을 받는데, 이 때 게시판의 현재 페이지(currPage)를 새로 만들어서 수정된 게시물의 제목을 화면에 보여준다.

㉔ '로그인' 상태에서 등록된 게시물을 삭제할 수 있으며, '삭제' 버튼이 클릭되면 실수로 삭제하는 것을 방지하기 위해서 "정말 삭제하시겠습니까?"라는 경고 창이 화면에 보여진다. '삭제' 버튼이 클릭되면 showAlertToDelete() 함수가 실행되어 경고 창이 브라우저의 중앙에 나타난다.

㉕ 삭제를 다시 물어보는 경고 창에서 '예' 버튼이 클릭되면 deleteArticle() 함수가 실행되어 서버의 데이터베이스에서 해당 게시물을 삭제한다. 삭제를 위해서 현재 Current 게시물Article의 id 값(currArticle)을 서버에 전달하고 'ajax_delete_article.php' 함수를 실행한다.

㉖ 삭제를 다시 물어보는 경고 창에서 '아니오' 버튼이 클릭되면 hidePopupAlert() 함수를 실행해서 경고 창을 화면에서 사라지게 한다.

㉗ 게시판의 모든 코드를 브라우저에서 다운받으면 자동으로 makeBoard(0) 함수를 실행해서 첫 번째 게시판의 내용을 서버로부터 받아서 화면에 보여준다.

㉘ 게시판의 모든 코드를 브라우저에서 다운받으면 '로그인'이 되었는지를 확인한 후에 '로그인' 상태이면 글을 작성하거나 편집할 수 있는 버튼을 화면에서 보여주고, 그렇지 않으면 페이지의 내용을 변경할 수 있는 버튼을 화면에서 보여주지 않는다.

또한, '로그인' 상태에서는 mousedown 이벤트를 동작할 수 있게 만들어서 입력 창에서 글을 작성하는 것이 가능하다.

[코드 5-21]은 서버의 데이터베이스에 저장된 게시판의 내용을 브라우저로부터 'Post 방식'으로 받은 'board'와 'page' 값에 따라 20개를 데이터를 JSON 형식으로 만들어서 전송하는 PHP 코드를 보여준다. 이때 'board'의 값은 데이터베이스의 하나의 테이블 이름과 같은데, 홈페이지에서 다수의 게시판들을 만든다면 게시판마다 데이터베이스 테이블을 만들어서 관리할 수 있다. 이 책에서는 'Story' 게시판을 위해서 데이터베이스의 'Story' 테이블을 사용하지만, 독자들은 이와 같은 방법으로 다양한 게시판 테이블을 만들어서 관리할 수 있다.

[코드 5-21] 게시판의 데이터(최대 20 개)를 보내는 PHP 코드 (/data/board/ajax_get_board.php)

```
<?php

require_once('../../library/php/config.php');
require_once('../../library/php/session.php');
require_once('../../library/php/mysql.php');

$board  = (isset($_POST['board'])) ? $_POST['board'] : "";                    //---①
$page   = (isset($_POST['page'])) ? $_POST['page'] : "";

$db = new MySQL();

if(!$db->connect($dbconfig)) {
        $db->close();
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

$sql = "SELECT id, subject, date, count FROM ".$board." ORDER BY id DESC LIMIT ".
           ($page*20).", 20";                                                //---②

$result = $db->query($sql);
if($result == false){
        $db->close();
        print '{ "error" : "데이터베이스 테이블에서 제목을 불러오지 못했습니다." }';
        exit();
}
$data = $db->fetchArray($result);

$sql = "SELECT COUNT(*) AS total FROM ".$board;                              //---③
```

```
$result = $db->query($sql);
if($result == false){
        $db->close();
        print '{ "error" : "데이터베이스 테이블에서 제목을 불러오지 못했습니다." }';
        exit();
}
$count = $db->fetchRow($result);

$db->close();

$strJSON = '{"boardPage":[';                                                    //---④
for ($i=0; $i<count($data); $i++) {
        if ($i != 0)
            $strJSON .= ',';
        $strJSON .= '{"id":'.$data[$i]["id"].',"subject":"'.$data[$i]["subject"].'","date":"'
                    .$data[$i]["date"].'","count":'.$data[$i]["count"].'}';
}
$strJSON .= '], "page":'.$page.', "count":'.$count["total"].'}';

print   $strJSON;
exit();

?>
```

① 게시판의 이름과 게시판의 페이지 번호를 'Post 방식'으로 받는다.

② 게시판에 보여지는 20개의 데이터를 가져오는 Query이며, Query의 내용은 게시판 테이블에서(FROM ".$board") id를 내림차순으로 정렬하고(ORDER BY id DESC) 20단위로 나누어진 페이지에서 해당 페이지의 20개 데이터들의(LIMIT ".($page*20).", 20) id, 제목, 날짜, 조회 수를 검색해서 가져온다.(SELECT id, subject, date, count)

③ 데이터베이스에 저장된 게시물들의 총 갯수를 가져오는 Query이며, Query의 내용은 게시판 테이블에서(FROM ".$board") 전체 게시물을 카운트하고 total 변수에 값을 넣는다.(SELECT COUNT(*) AS total)

④ 브라우저로 보내기 위한 데이터를 JSON 형식의 문자열로 만들어서 strJSON 변수에 저장한다. 게시판에 들어가는 20개의 데이터들은 boardPage의 값에 배열로 저장된다. 또한, 현재 페이지의 정수값은 page의 값에 저장하고, 데이터베이스에 저장된 전체 데이터의 갯수는 count의 값에 저장한다.

[코드 5-22]는 게시판에서 하나의 제목이 클릭되면 해당 게시물의 내용을 전송하는 서버의 PHP 코드를 보여준다. 데이터베이스의 내용을 조회해서 서버로 보내주는 단순과정일 수 있지만, 게시판의 모든 항목들은 조회 수를 가지고 있기 때문에 '로그

인'하지 않은 일반 사용자의 요청일 경우에만 조회 수를 증가시키는 과정을 추가하였다. 로그인에 상관없이 조회 수를 증가시키는 방법도 있지만, 여기서 만드는 게시판은 글을 쓴 사용자의 조회는 카운트하지 않기 위해서 이와 같이 구현하였다. 게시판에 글을 작성하고 난 이후에 다른 사람들이 얼마나 글을 읽었는지를 파악할 수 있는 좋은 방법이다.

[코드 5-22] 하나의 게시물의 제목과 내용을 전송하는 PHP 코드 (/data/board/ajax_get_article.php)

```
<?php

require_once('../../library/php/config.php');
require_once('../../libraryphp/session.php');
require_once('../../library/php/mysql.php');

$board  = (isset($_POST['board'])) ? $_POST['board'] : "";                    //---①
$id     = (isset($_POST['id'])) ? $_POST['id'] : "";

$ses = new session();
$db = new MySQL();

if(!$db->connect($dbconfig)) {
        $db->close();
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

if (!($ses->is_login())){                                                    //---②

        $db->startTransaction();

        $sql = "UPDATE ".$board." SET count = count + 1 WHERE id='".$id."'";

        $result = $db->query($sql);
        if($result == false){
            $db->rollback();
            $db->close();
            print '{ "error" : "데이터베이스 테이블에서 제목을 불러오지 못했습니다." }';
            exit();
        }

        $db->commit();
}

$sql = "SELECT subject, contents FROM ".$board." WHERE id='".$id."'";        //---③
```

```
$result = $db->query($sql);
if($result == false){
        $db->close();
        print '{ "error" : "데이터베이스 테이블에서 제목을 불러오지 못했습니다." }';
        exit();
}
$data = $db->fetchRow($result);

$db->close();

$strJSON = '{"article":{"subject":"'.$data["subject"].'","contents":"'.$data["contents"].'"}}';    //---④

print   $strJSON;
exit();

?>
```

---

① 게시판을 위한 데이터베이스 테이블 이름과 브라우저에 전송하기 위한 항목의 id를 'Post 방식'으로 받는다.

② 글을 읽으려는 사람이 '로그인' 상태가 아니라면 조회 수를 업데이트한다. 조회 수를 업데이트하는 Query의 내용은 게시판 테이블을 업데이트하는데(UPDATE ".$board.") id가 일치하는 데이터의(WHERE id='".$id"') 조회 수를 1 증가시킨다.(SET count = count + 1)

③ 게시판에서 클릭된 항목의 제목과 내용을 데이터베이스에서 가져오는 Query의 내용은 게시판 테이블에서(FROM ".$board.") id가 일치하는 데이터의(WHERE id='".$id."') 제목과 내용을 검색해서 가져온다.(SELECT subject, contents)

④ 브라우저로 전송하는 데이터는 JSON 형식의 문자열이며, article 항목의 값은 또 다른 JSON 형식으로 subject와 contents 항목을 만들어서 제목과 내용을 값으로 저장한다.

[코드 5-23]은 사용자가 게시판에 글을 작성해서 저장할 때 서버에서 데이터베이스에 저장하는 PHP 코드를 보여준다. 서버에 저장할 때는 저장을 시도하는 사용자가 '로그인' 상태인지를 확인하고, 브라우저로부터 받은 내용을 데이터베이스에 저장하는 간단한 구조로 이루어졌다. 데이터베이스에 저장할 때에는 저장되는 날짜를 같이 넣어주어야 하는데, 현재 시간을 초단위로 알려주는 PHP의 time() 함수와 초단위의 시간을 날짜로 만들어주는 date() 함수를 사용한다.

[코드 5-23] 사용자가 작성한 글을 서버에 저장하는 PHP 코드 (/data/board/ajax_save_article.php)

```php
<?php

require_once('../../library/php/config.php');
require_once('../../library/php/session.php');
require_once('../../library/php/mysql.php');

$ses = new session();

if(!($ses->is_login())){                                                    //---①
        print '{ "error" : "로그인되지 않았습니다." }';
        exit();
}

$board  = (isset($_POST['board'])) ? $_POST['board'] : "";                  //---②
$title          = (isset($_POST['title'])) ? $_POST['title'] : "";
$contents = (isset($_POST['contents'])) ? $_POST['contents'] : "";

$curDate = date("Y-m-d", time());                                           //---③

$db = new MySQL();

if(!$db->connect($dbconfig)) {
        $db->close();
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

$db->startTransaction();

$sql = "INSERT INTO ".$board." VALUES (0, '".$title."', '".$contents."', '".$curDate."', 0)"; //---④

$result = $db->query($sql);
if($result == false){
        $db->rollback();
        $db->close();
        print '{ "error" : "데이터베이스 테이블에서 등록되지 않았습니다." }';
        exit();
}

$db->commit();

$db->close();

print   0;
exit();
```

```
?>
```

① '로그인' 상태인지를 확인해서 '로그인' 상태가 아니면 저장을 하는 과정 없이 프로그램을 종료한다. 브라우저에서 로그인 상태가 아니면 글을 작성하는 환경을 화면에서 볼 수 없지만, 네트워크로 비정상 데이터를 보내는 사람도 있을 수 있기 때문에 데이터베이스의 내용을 변경할 때는 항상 정상적인 세션이 연결되었는지를 확인하는 것이 좋다.

② 게시판을 위한 데이터베이스 테이블 이름과 저장하기 위한 글의 제목과 내용을 'Post 방식'으로 받는다.

③ 현재 날자를 $curDate 변수에 저장하는데, time() 함수는 함수가 실행되는 시간을 초단위의 정수형으로 만들어 주고 date() 함수는 초단위의 시간에서 "연-월-일"의 현재 날짜를 만든다.

④ 서버의 데이터베이스에 사용자가 작성한 글을 삽입하는 Query로서 Query의 내용은 게시판 테이블에 추가하는데(INSERT INTO ".$board.") id는 자동으로 증가하는 값이기 때문에 0을 넣고, '조회수'는 0으로 초기화 하고, '제목'과 '내용'과 '현재 날짜'의 값들을 넣는다. (VALUES (0, '".$title."', '".$contents."', '".$curDate."', 0))

[코드 5-24]는 서버의 데이터베이스에 저장된 내용을 업데이트하는 PHP 코드이며, [코드 5-23]에서 저장을 하는 코드와 대부분 동일하다. 글을 수정하는 과정은 수정하는 항목의 id를 브라우저로부터 받기 때문에 데이터베이스에서 id가 같은 데이터를 새로운 내용으로 업데이트하면 된다. 브라우저에서 '수정' 버튼이 클릭되면 실행되는 서버의 프로그램이고 데이터베이스의 내용이 변경되기 때문에 '로그인' 상태인지를 확인하는 과정이 가장 먼저 수행된다.

[코드 5-24] 게시글을 수정하는 서버의 PHP 코드 (/data/board/ajax_update_article.php)

```
<?php

require_once('../../library/php/config.php');
require_once('../../library/php/session.php');
require_once('../../library/php/mysql.php');

$ses = new session();

if(!($ses->is_login())){
        print '{ "error" : "로그인되지 않았습니다." }';
        exit();
}
```

```
$db = new MySQL();

$board  = (isset($_POST['board'])) ? $_POST['board'] : "";
$id       = (isset($_POST['id'])) ? $_POST['id'] : "";
$title              = (isset($_POST['title'])) ? $_POST['title'] : "";
$contents = (isset($_POST['contents'])) ? $_POST['contents'] : "";

$curDate = date("Y-m-d", time());

if(!$db->connect($dbconfig)) {
        $db->close();
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

$db->startTransaction();

$sql = "UPDATE ".$board." SET subject='".$title."', contents='".$contents."' WHERE id=".$id;
                                                                                //---①
$result = $db->query($sql);
if($result == false){
        $db->rollback();
        $db->close();
        print '{ "error" : "데이터베이스 테이블에서 업데이트되지 않았습니다." }';
        exit();
}

$db->commit();

$db->close();

print   0;
exit();

?>
```

① 데이터베이스의 내용을 수정하는 Query의 내용은 게시판 테이블을 업데이트하는데(UPDATE ".$board.") id가 일치하는 데이터의(WHERE id=".$id) 제목과 내용을 변경한다.(SET subject='".$title."', contents='".$contents."')

[코드 5-25]는 데이터베이스에 저장된 게시판의 게시물 하나를 삭제하는 서버의 PHP 코드이며, 데이터베이스의 내용을 변경하는 과정이 진행되기 때문에 정상적으로 세션이 이루어졌는지를 먼저 확인하고 데이터를 삭제한다. 삭제를 위해서는 데이터베이스 테이블의 이름과 삭제하기 위한 데이터의 id를 브라우저로 받아서 해당 id

를 가지고 데이터를 삭제한다.

[코드 5-25] 서버에 저장된 게시글 하나를 삭제하는 PHP 코드 (/data/board/ajax_delete_article.php)

```
<?php

require_once('../../library/php/config.php');
require_once('../../library/php/session.php');
require_once('../../library/php/mysql.php');

$ses = new session();

if(!($ses->is_login())){
        print '{ "error" : "로그인되지 않았습니다." }';
        exit();
}

$board  = (isset($_POST['board'])) ? $_POST['board'] : "";
$id     = (isset($_POST['id'])) ? $_POST['id'] : "";

$db = new MySQL();

if(!$db->connect($dbconfig)) {
        $db->close();
        print '{ "error" : "데이터베이스 연결에 실패했습니다." }';
        exit();
}

$db->startTransaction();

$sql = "DELETE FROM ".$board." WHERE id=".$id;                    //---①

$result = $db->query($sql);
if($result == false){
        $db->rollback();
        $db->close();
        print '{ "error" : "데이터베이스 테이블에서 지워지지 않았습니다." }';
        exit();
}

$db->commit();

$db->close();

print   0;
exit();

?>
```

① 데이터베이스에 저장된 내용을 삭제하는 Query는 간단한데, 데이터베이스에서 Primary 키로 사용되는 id를 가진 데이터를 삭제하기만 하면 된다. Query의 내용은 게시판 테이블에서 id가 일치하는 데이터를 삭제한다.(DELETE FROM ".$board." WHERE id=".$id)

지금까지 만든 게시판의 기본 골격은 태그만을 사용했는데, 태그만으로 HTML의 많은 태그들을 대체하는 것이 가능하다는 것을 설명하기 위해서였다. 태그를 사용하지 않고 기존의 HTML 태그를 사용하는 방법은 현재 인터넷에서 보는 대부분의 게시판들인데, [코드 5-26]은 게시판을 만들기 위해서 <table></table> 태그를 사용하는 방법을 보여준다. [코드 5-26]은 [코드 5-20]과 대부분 같으며, 단지 게시판을 만드는 makeBoard() 함수에서 <table></table> 태그를 사용해서 게시판을 만드는 부분이 다르다. 사각형을 만드는 태그를 <table></table> 태그 안에서 <tr></tr>, <th></th>, <td></td> 태그들로 변경된 것을 볼 수 있으며, CSS를 사용해서 테이블의 디자인 속성을 변경하였다.(CSS 디자인 속성을 위한 코드는 'smenu_3_4.html' 파일 안에서 확인할 수 있다.)

[코드 5-26] <table></table> 태그로 게시판을 만드는 JavaScript (/kr/data/container/smenu_3_4.html)

```
##### 생략 #####

    function makeBoard(noPage) {

        var param = { board: 'Story', page: noPage };
        var jsonData = doAjax('../data/board/ajax_get_board.php', param);

        var obj = JSON.parse(jsonData);

        currPage = obj.page;

        var no = obj.count - (obj.page * countInPage);

        var str = '<table>';                                                    //---①
        str += '<tr id="boardHead"> \
                        <th id="boardHeadColumn1">번호</th> \
                        <th id="boardHeadColumn2">제목</th> \
                        <th id="boardHeadColumn3">날짜</th> \
                        <th id="boardHeadColumn4">조회수</th>';
```

```
        str += '</tr>';

        for (var i=0; i<obj.boardPage.length; i++) {

                str += '<tr class="boardRow">';
                str += '<td class="boardColumn1">'+no+'</td> \
                  <td class="boardColumn2" onclick="showArticle('+obj.boardPage[i].id+');"> \
                    '+setQuotaionMark(obj.boardPage[i].subject)+'</td> \
                  <td class="boardColumn3">'+obj.boardPage[i].date+'</td> \
                  <td class="boardColumn4">'+obj.boardPage[i].count+'</td>';
                str += '</tr>';

                no -= 1;
        }
        str += '</table>';

        str += '<div style="position:relative;top:10px;left:200px;width:300px;height:30px;">';
        if (obj.page > 0) {
            str += '<div class="button2" onclick="makeBoard('+(obj.page - 1)+');"> << </div>';
        }
        var startIndex = obj.page - 4;
        if (startIndex < 0) {
                startIndex = 0;
        }
        var lastIndex = Math.ceil(obj.count / countInPage);
        var endIndex = obj.page + 5;
        if (endIndex > lastIndex) {
                endIndex = lastIndex;
        }
        for (var i=startIndex ; i<endIndex; i++) {
                if (i == obj.page) {
                    str += '<div class="bold black" style="width:20px;float:left;">'+ (i+1) +'</div>';
                }
                else {
                   str += '<div class="button2" onclick="makeBoard('+i+');">'+ (i+1) +'</div>';
                }
        }
        if (no > 0) {
            str += '<div class="button2" onclick="makeBoard('+(obj.page + 1)+');"> >> </div>';
        }
        str += '</div><br><br>';

        $("#board").empty().html(str);
    }
```

생략

---

① 게시판에 들어갈 객체의 내용들을 문자열로 만들어서 str 변수에 저장한다. 게시판의 표Table는 <table></table> 객체를 사용해서 만들었으며, <table></table> 객체에 대한 사용법은 1.1장의 마지막에 설명하였다.

이번 장에서는 웹 프로그래밍에서 가장 중요하게 생각하는 데이터를 보관하기 위해서 데이터베이스를 사용해서 관리하는 방법을 설명하였는데, 쉬운 이해를 위해서 최대한 간단하게 구현하였다. 실생활에서 접하는 대부분의 웹 서비스들은 훨씬 복잡한 구조를 만드는데, 이 책을 이해한 독자는 시간을 들여 프로그래밍을 진행하기 바란다. 이전 Chpater의 '웹 블로그'처럼 단순히 정보만을 제공하는 사이트는 서비스를 공격하는 사람들이 거의 없지만, 다수의 사용자 정보를 가지고 있는 시스템을 공격하는 사람들은 생각보다 많이 있다. 그래서, 데이터베이스의 내용을 변경할 수 있는 코드는 정상적인 로그인을 통해서 세션을 연결해야만 한다. 많은 서비스에서 회원가입이나 로그인을 할 때 보안문자를 입력하게 만드는데, 인터넷에서는 프로그램을 사용해서 자동으로 회원가입을 하고 로그인을 해서 다수의 게시물을 올리는 프로그램을 사용하는 사람들이 꽤 존재한다. 웹 서비스를 운영하는 많은 사람들은 나쁜 쪽으로 지식을 사용하는 사람들로부터 시스템을 보호하기 위해서 다양한 방법으로 시스템을 보호하고 있다. 또한, 네트워크를 조금 알게 되면 서버와 주고받는 데이터Packet Date의 내용을 확인할 수 있는데, 네트워크에 대해서 조금 알고 있다고 자신을 대단하게 생각하며 악성 패킷들을 보내는 한심한 사람들이 의외로 많다. 그들은 자신들을 스스로 '해커'라고 부르지만, 돈을 주면 무엇이든 하는 현대판 "인터넷 낭인"이라 부르고 싶다. 대다수의 멋진 '해커'들은 자신의 기술을 계속 발전시키며 인터넷 세상을 지키는 반면, 소수의 "인터넷 낭인"들이 인터넷 세상을 어지럽히고 있는 상황이다.

chapter 6

Javascript 프로그래밍

웹 사이트를 만들면서 사용자들에게 좀더 참신하고 오래 머무를 수 있는 환경을 제공하려는 마음은 웹 프로그래머의 바람이다. 웹 사이트를 동적으로 움직이게 만들기 위해서 다양한 방법들이 연구되어 왔는데, 효과적인 웹 사이트를 만들기 위해서 가장 좋은 방법은 JavaScript 언어로 프로그래밍을 하는 것이다. 웹 프로그래밍에서는 서버쪽 프로그래밍을 하는 사람들을 더 수준 높게 보는 편인데, 그에 못지 않게 JavaScript 언어로 브라우저의 화면을 화려하게 만드는 것도 아주 중요하다. 그리고, JavaScript 언어로 만든 코드는 누구나 '소스보기' 기능을 사용해서 코드의 내용을 볼 수 있기 때문에 누구나 쉽게 복사할 수 있지만, 오픈소스의 개념으로 많은 사람들에게 정보를 공유하고 있으니 좀더 자유롭게 프로그래밍을 할 수 있다. 이번 장에서는 브라우저에서 재미있는 효과를 줄 수 있는 간단한 JavaScript 프로그램들을 설명하는데, 그동안 필자가 JavaScript 언어를 공부하면서 만들었던 코드들이다. 독자들은 이런 방법들을 응용해서 더 멋진 코드들을 만들기를 바란다. 대부분의 웹 프로그래머들은 JavaScript 언어에서 단순한 몇 가지만을 반복해서 사용하는 경향이 있는데, JavaScript 언어는 다른 프로그래밍 언어에서 제공하는 모든 연산자를 제공한다. 예를 들면, 비트Bit 연산자의 경우 일반적인 웹 프로그래밍을 하면서 잘 사용하지 않지만 암호화 프로그래밍도 가능하다. 그러나, 브라우저에서만 동작하고 코드를 읽을 수 있는 스크립트Script 언어이기 때문에 보안성이 없어서 사용을 할 수 없을 뿐이다. JavaScript는 Java와는 전혀 다른 프로그래밍 언어지만, Java라는 단어가 들어간 이름에서 유추할 수 있는 것은 처음 만들어질 때 Java 언어를 기본으로 해서 만들었을 것 같다. 웹 프로그래밍을 하다보면 서버에서 동작하는 JSP나 PHP를 사용하는 것이 더 어렵게 느껴질 수 있지만, 사용자는 브라우저에서 보여지는 화면만을 보기 때문에 JavaScript 언어로 사용자 친화적인 화면을 만드는 것도 매우 중요하다.

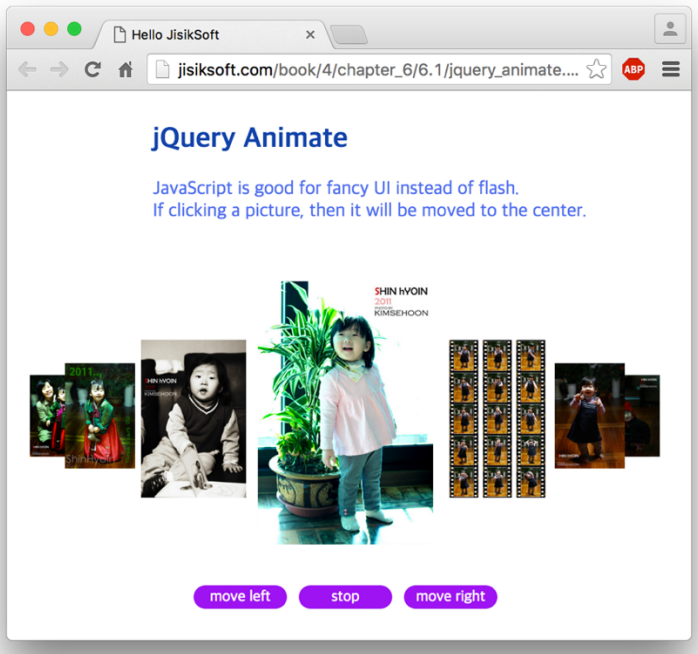
6.1 jQuery Animate

jQuery의 animate() 함수를 사용하면 동적으로 움직이는 화면을 만들기가 쉽다. 오래 전에는 화장품이나 자동차 회사의 사이트에서 화려한 화면을 만드는 데 Flash를 많이 사용했다. 그러나, 현재는 JavaScript 언어로 다양한 기능을 하는 라이브러리들이 많이 생겨서 Flash를 사용하지 않고도 사용자들의 이벤트를 처리할 수 있는 화려한 사이트를 만들 수 있게 되었다.

[그림 6-1]은 12개의 사진들이 좌우로 움직이는 것을 보여주는데, 사진의 디자인 속성을 변경해서 입체적으로 움직이는 효과를 준 것이다. 이 때 사용한 함수는 jQuery의 animate() 함수인데, 2초 동안 12개의 사진이 설정된 크기와 위치로 서서히 변경되며 입체적인 효과를 준다. 또한 브라우저는 평면이지만 브라우저 안에서 보이는 사진들은 CSS의 z-index 속성을 사용해서 겹쳐지는 사진들의 높낮이를 설정한다. 평면은 (x, y)라는 두 개의 값으로 좌표가 정해지는데, 'z-축'을 만들어서 화면 위쪽으로 HTML 태그 객체의 위치를 조정해서 평면에서 입체적인 화면을 만들 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_6/6.1/jquery_animate.html

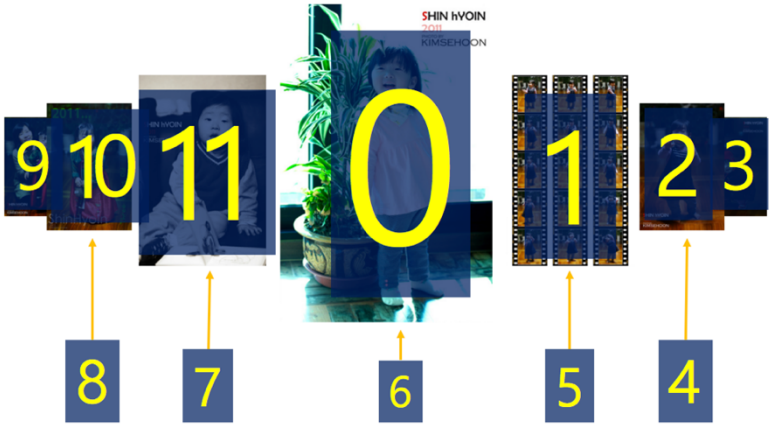
그림 6-1 12 개의 사진들이 회전하는 웹 프로그램



회전하는 효과를 주기 위해서 12개 사진들의 디자인 속성을 변경해야 한다. 12개의 디자인 속성을 정한 후에 사진을 포함하는 태그들의 디자인을 jQuery

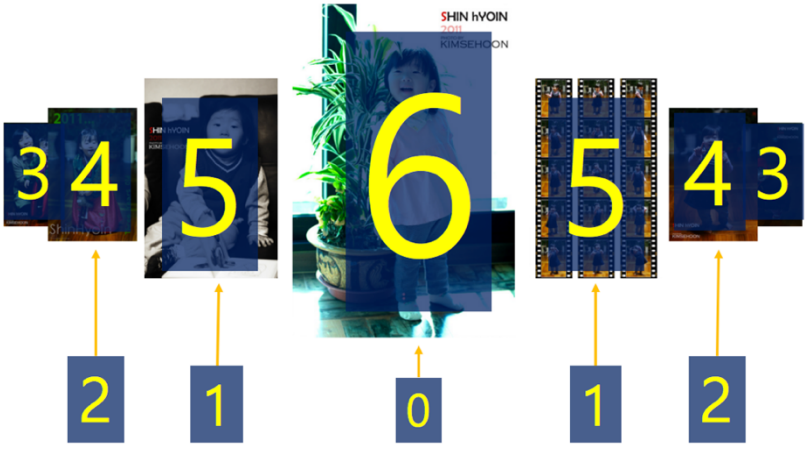
의 animate() 함수를 사용해서 정해진 시간 동안에 CSS 디자인 값들을 변경한다. [그림 6-2]는 12개의 숫자로 순서를 정해서 모든 사진들의 디자인 속성값을 차례대로 조정해서 사진의 위치를 변경한다. 디자인 속성들은 마치 사진이 입체적으로 배열된 효과를 준다. 이와 같이 배열된 상태에서 태그들의 위치와 크기 및 z-index들을 변경하면 모든 사진이 입체적으로 동시에 움직이는 효과를 갖는다.

그림 6-2 위치 속성을 순서대로 만들어서 사진의 디자인 속성을 변경한다.



입체적인 효과를 얻기 위해서는 앞쪽에 위치한 사진과 뒷쪽에 위치한 사진을 정해야 하는데, 이때 z-index를 사용한다. CSS 디자인의 z-index 속성에 값을 주게되면 큰 값이 앞쪽에 위치한 것으로 보이는데, [그림 6-3]은 z-index의 값을 사용해서 원형의 이미지 위치를 정한 것이다. 맨앞쪽의 사진은 z-index 값으로 6을 가지는데, 뒤에 숨겨진 사진의 z-index 값이 0이기 때문에 보이지 않는다. 동적으로 동작하는 웹 프로그램의 경우에는 z-index에 값을 부여해서 태그 객체들을 입체적으로 보이게 할 수 있다. 만약 z-index를 사용하지 않으면 종속된 태그 객체에서는 상위Parent 태그의 객체위에 자식Child 태그의 객체들이 위치한다.

그림 6-3 'z-index' 속성으로 사진을 앞쪽에서 보이게 만든다.



동적인 웹 프로그램은 브라우저에서 자동으로 동작하기 때문에 JavaScript 언어를 사용해야만 한다.

[코드 6-1]은 입체적으로 움직이는 사진들을 위한 HTML 코드를 보여주고, [코드 6-2]는 자신들의 디자인 속성들을 정의하고 animate() 함수를 사용해서 동적으로 동작하는 화면을 만드는 JavaScript 코드를 보여준다.

[코드 6-1] 입체적으로 움직이는 사진들을 보여주기 위한 HTML 코드 (jquery_animate.html)

```
##### 생략 #####

<body>
    <br><div id="subject">jQuery Animate</div><br>
    <div id="contents">JavaScript is good for fancy UI instead of flash.<br>
       If clicking a picture, then it will be moved to the center.<br>
    </div>
    <div id="picture_area"></div>                                          //---①

    <div style="position:relative;left:155px;top:10px;">                   //---②
       <div class="button" onclick="move(1);">move left</div>
       <div class="button" style="left:10px;" onclick="move(0);">stop</div>
       <div class="button" style="left:20px;" onclick="move(2);">move right</div>
    </div>

    <script>
       $('#picture_area').html(getImages());                               //---③

       move(1);                                                            //---④
    </script>
</body>
</html>
```

① 12개의 사진 객체들은 'picture_area'라는 id 값을 가진 객체에 추가되는데, 프로그램의 모든 코드를 브라우저에서 다운받으면 JavaScript 코드가 실행되어서 자동으로 12개의 사진 객체들을 넣어준다.

② 화면에는 3개의 버튼들이 존재하는데, 사진을 왼쪽으로 움직이게 하는 'move left' 버튼, 오른쪽으로 움직이게 하는 'move right' 버튼, 그리고 동작을 멈추게 하는 'stop' 버튼이 있다. 모든 버튼은 클릭되면 move() 함수를 실행시키며 매개변수로 전달되는 숫자에 의해서 동작이 결정된다.

③ 브라우저에 모든 코드들을 다운받으면 자동으로 실행되는 JavaScript 코드이며, getImages() 함수를 실행해서 12개 사진들의 객체들을 문자열로 만들어서 'picture_area'라는 id 값을 가진 객체에 넣어준다.

④ 브라우저에 모든 객체들이 보여지면 move() 함수를 사용해서 사진들을 왼쪽으로 계속 움직이게 만든다.

[코드 6-2] animate() 함수를 사용해서 움직이는 화면을 만드는 JavaScript 코드 (/js/jquery_animate.js)

```
var timerMove = null;                                                    //---①
var curIndex = 0;                                                        //---②

var valPos = [ //z-index, top, left, width                               //---③
                [ 6, 50, 210, 150 ], [ 5, 100, 370, 90 ], [ 4, 120, 464, 60 ],
                [ 3, 130, 507, 47 ], [ 2, 135, 400, 42 ], [ 1, 140, 370, 40 ],
                [ 0, 150, 275, 30 ], [ 1, 140, 145, 40 ], [ 2, 135, 50, 42 ],
                [ 3, 130, 15, 47 ], [ 4, 120, 45, 60 ], [ 5, 100, 110, 90 ]
            ];

var pos = [ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 ];                      //---④
var temp = [ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 ];                     //---⑤

function movePicture(id) {                                               //---⑥

    var prefix = id.substring(0, 1);                                     //---⑦
    var numId = id.substring(1) * 1;                                     //---⑧
    var direction = 0;                                                   //---⑨
    var gap = 0;                                                         //---⑩

    curIndex = numId;

    if (pos.indexOf(numId) > 3 && pos.indexOf(numId) < 9) {              //---⑪
        return false;
    }
    if (pos.indexOf(numId) < 4) {                                        //---⑫
        direction = -1;
        gap = pos.indexOf(numId);
    }
    if (pos.indexOf(numId) > 8) {                                        //---⑬
        direction = 1;
        gap = 12 - pos.indexOf(numId);
    }

    for (var i = 0; i < gap; i++) {                                      //---⑭
        for (var j = 0; j < 12; j++) {
```

```
                if (direction === 1) {                                  //---⑮
                        temp[j] = pos[(j + 11) % 12];
                } else {
                        temp[j] = pos[(j + 1) % 12];
                }
                $("#" + prefix + temp[j]).animate({                     //---⑯
                        "z-index" : valPos[j][0],
                        "top" : valPos[j][1]+"px",
                        "left" : valPos[j][2]+"px",
                        "width" : valPos[j][3]+"px"
                }, 1000);
        }

        for (var j = 0; j < 12; j++) {                                  //---⑰
                pos[j] = temp[j];
        }
    }
};

function moveLeft() {                                                   //---⑱
    curIndex += 1;
    curIndex = curIndex % 12;
    movePicture('p' + curIndex);
}

function moveRight() {                                                  //---⑲
    curIndex += 11;
    curIndex = curIndex % 12;
    movePicture('p' + curIndex);
}

function move(action) {                                                 //---⑳
    clearTimeout(timerMove);

    if (action == 1) {
        moveLeft();
        timerMove = setInterval("moveLeft()", 2000);
    } else if (action == 2) {
        moveRight();
        timerMove = setInterval("moveRight()", 2000);
    }
}

function getImages() {                                                  //---㉑
    var str = '';
```

```
    for (var i = 0; i < 12; i++) {
        str += '<div id="p'+i+'" style="position:absolute;'
                        +'z-index:'+valPos[i][0]+';'
                        +'top:'+valPos[i][1]+'px;'
                        +'left:'+valPos[i][2]+'px;'
                        +'width:'+valPos[i][3]+'px;"'
                        +' onclick="movePicture(this.id)">'
                        +'<img src="./img/'+(i+1)+'.jpg" style="width:100%;"></img>'
                        +'</div>';
    }
    return str;
}
```

① 사진들을 자동으로 움직이게 할 때 setInterval() 함수를 사용해서 계속적으로 반복되는 동작을 만드는데, setInterval() 함수가 실행될 때 반환되는 값을 timerMove 변수에 저장한다. 동작을 멈추거나 다른 방향으로 움직이게 하기 위해서는 timerMove 변수에 설정된 값을 사용해서 반복적으로 실행되는 동작을 멈추어야 하는데, 이때 clearTimeout(timerMove) 함수를 실행한다.

② 12개의 사진들은 숫자를 포함한 id 값을 갖는데, 입체적인 사진의 중앙에 어떤 사진이 위치해 있는지를 알기 위해서 현재 중앙에 위치한 사진의 번호를 curIndex 변수에 저장한다. 모든 사진들의 현재 위치를 curIndex 변수를 통해서 알고 있기 때문에 모든 사진들의 위치를 포함한 디자인을 변경하는 것이 가능하다.

③ 입체적으로 위치한 사진들의 디자인 값들을 배열로 갖는데, 하나의 배열는 'z-index', 'top', 'left', 'width' 등의 디자인 속성값들을 포함하고 있다. 사진이 변경되는 것은 위치가 변하기 때문에 'top'과 'left' 값에 변화를 주는 것이며, 사진의 크기가 변하기 위해서 'width' 값이 필요하다. 또한 입체적인 사진의 배열을 위해서 앞쪽에 있는 사진의 'z-index' 값이 높으며, 뒤쪽에 있는 사진들의 'z-index' 값들이 순차적으로 작아진다.

④ 12개의 사진들의 디자인 값들을 기억하기 위해서 사용하는 방법으로 pos 배열이 사용되는데, 배열의 순서는 사진의 id 값과 같고 배열에 저장된 값은 valPos 배열에 저장된 디자인 속성값들의 순서를 의미한다. 예를 들면, [그림 6-2]와 같이 12개의 사진들은 순서를 가지고 있는데, 현재 중앙에 위치한 사진의 id 값이 3이라고 가정하자. 그렇다면 pos 배열에는 [9, 10, 11, 0, 1, 2, 3, 4, 5, 6, 7, 8]라는 값이 저장되는데, "pos[3] = 0"으로 설정되었음을 알 수 있기 때문에 id 값이 3인 사진의 디자인 속성값은 'valPos[0]'의 값을 갖는다.

⑤ temp 배열은 pos 배열을 위한 임시저장소로 사용하기 위해서 사용하는 배열이며, 클릭된 사진을 이동할 때 순차적으로 변경되는 pos의 값들을 임시로 저장한다.

예를 들면, 현재 "pos = [9, 10, 11, 0, 1, 2, 3, 4, 5, 6, 7, 8]"이라고 가정할 때 사진이 왼쪽으로 이동하면 "temp = [10, 11, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9]"의 값을 갖고, 오른쪽으로 이동하면 "temp = [8, 9, 10, 11, 0, 1, 2, 3, 4, 5, 6, 7]"의 값을 임시로 저장한다.

⑥ 사진이 클릭되거나 좌우로 움직일 때 실행되는 함수로서, 12장 사진들의 위치를 계산하고 jQuery의 animate() 함수를 사용해서 움직이는 효과를 만든다. 함수의 매개변수로 받는 id는 중앙에 위치해야 하는 사진 태그의 id 값을 받는데, 예를 들면 클릭된 사진의 id 값이 'p3'이라면 'movePicture('p3')'이라는 함수가 실행되고 해당 사진이 중앙에 위치하게 사진들이 이동한다.

⑦ HTML 태그의 클래스나 id 값은 숫자로 시작하는 값이 될 수 없기 때문에, 모든 사진 객체의 id 값은 'p' 문자로 시작한다. 예를 들면 첫 번째 사진의 id 값은 'p0'이고, 그 다음 사진은 'p1'이라는 값을 갖는다. 여기서 prefix 변수는 사진의 id 값에서 첫 번째 문자인 'p'를 저장한다.

⑧ 사진 태그에 부여된 숫자를 가져와서 numId 변수에 저장한다. 숫자 '1'을 곱해서 문자열로 만들어진 숫자를 정수형으로 변경한다.

⑨ direction 변수는 사진이 회전하는 방향을 나타내는데, direction 변수의 값이 '-1'이면 사진들이 왼쪽으로 이동하고 '1'이면 오른쪽으로 이동한다.

⑩ gap 변수는 이동하는 횟수를 정하기 위해서 사용되는 변수로서, 클릭된 사진이 중앙에 위치하기 위해서 회전 해야하는 횟수를 값으로 갖는다.

⑪ 클릭한 사진이 뒤에 위치했으면 아무런 동작도 하지 않는다. 즉, 앞에 보이는 사진이 아니라 뒤에 있으면 디자인을 위한 값이라면 4, 5, 6, 7, 8 값들 중에 하나이다.

⑫ 클릭한 사진의 디자인 위치의 값이 1, 2, 3 값 중에 하나이면 중앙의 오른쪽에 위치한 사진이기 때문에 왼쪽으로 회전하기 위해서 direction 변수는 '-1'을 저장한다. 클릭한 사진이 중앙에 오기 위해서는 회전해야 하는 횟수를 gap 변수에 저장한다.

⑬ 클릭한 사진의 디자인 위치의 값이 9, 10, 11 값 중에 하나이면 중앙의 왼쪽에 위치한 사진이기 때문에 오른쪽으로 회전하기 위해서 direction 변수는 '1'을 저장한다. 클릭한 사진이 중앙에 오기 위해서는 회전해야 하는 횟수를 계산해서 gap 변수에 저장한다.

⑭ 회전해야 하는 횟수가 gap 변수에 저장되었기 때문에 for loop을 사용해서 gap의 숫자만큼 반복한다. 또한 한번 회전할 때는 12개의 사진들을 동시에 이동해야 하기 때문에 for loop을 사용해서 12회를 반복한다.

⑮ direction 변수의 값이 '1' 이면 오른쪽으로 이동하고 '-1'이면 왼쪽으로 이동하기 때문에 pos 배열의 값을 방향에 맞추어 하나씩 이동한 후의 값을 temp 배열에 저장한다.

⑯ 사진 객체를 animate() 함수를 사용해서 디자인 설정 값들을 변경하며, 1초(1000ms)동안 설정 값들이 조금씩 변하기 때문에 화면에서 움직이는 효과를 보게된다. 'animate()' 함수의 매개변수로 CSS 디자인 속성들이 JSON 형식으로 입력되는 것을 알수 있다.

⑰ 변경되는 디자인을 위한 값들의 임시로 저장한 temp 변수의 값들을 pos 변수에 차례대로 넣어준다.

⑱ 사진을 왼쪽으로 한번 회전할 때 실행하는 함수로서, 왼쪽으로 계속 움직이는 효과는 이 함수를 계속해서 반복적으로 사용하는 것이다.

⑲ 사진을 오른쪽으로 한번 회전할 때 실행하는 함수로서, 오른쪽으로 계속 움직이는 효과는 이 함수를 계속해서 반복적으로 사용하는 것이다.

⑳ 화면에서 사진의 회전을 위해서 만들어진 버튼이 클릭될 때 실행되는 함수로서, 매개변수인 action의 값이 1이면 왼쪽으로 계속 회전하고 2이면 오른쪽으로 회전한다. 함수가 실행되었을 때는 항상 이전에 실행된 setInterval() 함수를 종료하기 위해서 clearTimeout() 함수가 실행되기 때문에, 1 또는 2 이외의 값을 매개변수로 받으면 회전을 멈춘다.

㉑ 12개 사진들을 화면에 보여주기 위해서 12개의 태그 객체들을 만드는 함수로서 문자열로 만들어진 객체들을 반환한다. 공통적인 패턴Pattern을 가지고 다수의 HTML 객체들을 만들때 JavaScript 언어를 사용해서 이와 같이 만들면, HTML 파일에서 많은 내용을 작성하지 않아도 되기 때문에 많은 사람들이 자주 사용하는 방법이다.

6.2 jQuery Draggable

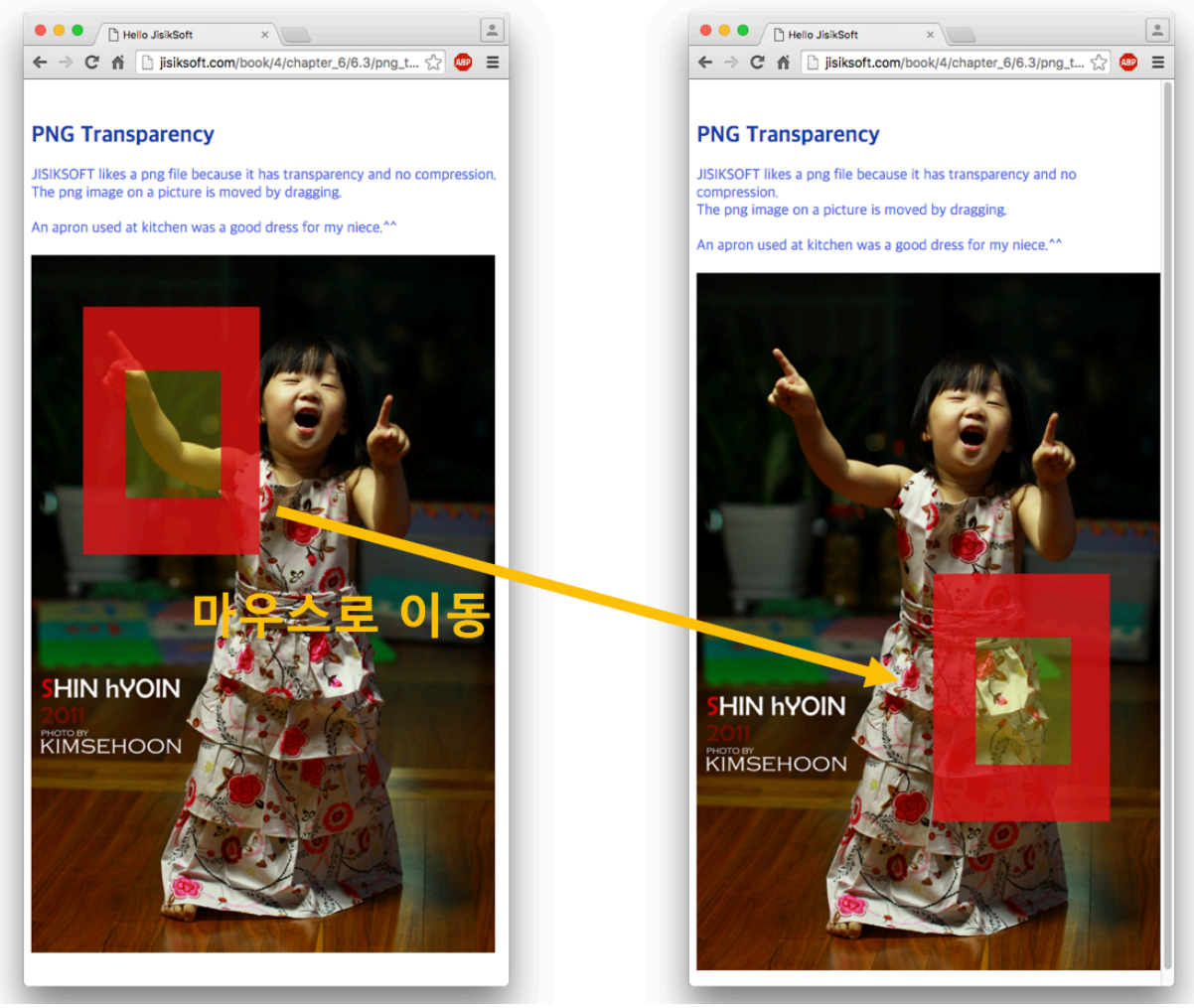
이번 장에서는 jQuery의 draggable() 함수를 사용해서 이미지를 마우스로 이동할수 있는 웹 프로그램의 예제를 보여준다. 여기서 중요한 것은 jQuery의 함수들은 매개변수를 사용해서 다양한 효과를 낼 수 있는데, 단순히 이미지를 이동하는 것은 draggable() 함수만 사용하면 되고, 특정 영역 안에서만 움직이게 만드는 함수도 있

다.

[그림 6-4]는 사진 위의 이미지를 이동할 수 있는 웹 페이지를 보여주는데, Popup으로 만들어진 이미지는 사진 안에서만 이동이 가능하다. Popup으로 만들어진 이미지는 PNG 파일의 이미지이며, PNG 형식의 이미지에 투명도를 주어서 '셀로판지'와 같은 효과를 주었다. PNG 형식의 이미지는 다른 형식의 이미지보다 파일의 용량이 크지만, 투명도를 줄 수 있기 때문에 웹 프로그래밍에서 다양하게 사용할 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_6/6.2/jquery_draggable.html

그림 6-4 Popup 이미지를 사진 안에서 마우스로 이동시키는 화면



[코드 6-3]은 마우스로 이미지를 이동시키기 위해서 jQuery의 draggable() 함수를 사용하는 코드를 보여주는데, draggable() 함수의 속성 중의 하나인 containment를 사용해서 특정 객체 안에서만 이미지가 이동할 수 있게 만들었다. 웹 프로그래밍에서 하나의 태그가 하나의 객체라는 것을 이해하고 나면, 객체의 클래스나 id 값을 가지고 객체에 접근할 수 있기 때문에 'containment' 속성에 객체에 접근할 수 있는

id 값을 넣으면 draggable() 함수 안에서 해당 객체에 접근해서 객체의 영역 안에서만 이미지를 이동하게 코드로 만들어졌음을 짐작할 수 있다.

[코드 6-3] 마우스의 드래그로 이미지를 움직이는 HTML 과 JavaScript 코드 (jquery_draggable.html)

```
##### 생략 #####
<body>
<br><div id="subject">PNG Transparency</div><br>
<div id="contents">JISIKSOFT likes a png file because it has transparency and no compression.<br>
The png image on a picture is moved by dragging.<br><br>
An apron used at kitchen was a good dress for my niece.^^</div><br><br>
<div id="picture_1" style="width:500px;">                                    //---①
    <img src="./img/hyoin.jpg">
    <div id="popup">                                                         //---②
        <img src="./img/popup.png">
    </div>
</div>
<br>
<script>
    $("#popup").draggable({ containment:'#picture_1' });                     //---③
</script>

</body>
</html>
```

① 화면에 사진을 보여주기 위해서 'picture_1'이라는 id 값을 가진 태그를 만들었다.

② 화면에 Popup 이미지를 보여주기 위해서 'popup'이라는 id 값을 가진 태그를 만들었다.

③ jQuery의 draggable() 함수를 사용해서 'popup'이라는 id 값을 가진 객체를 마우스로 이동할 수 있게 만들었다. 이때 draggable() 함수의 매개변수를 사용해서 다양한 기능을 설정할 수 있는데, 'containment' 속성에 사진을 위해 만들어진 'pictuer_1'이라는 id 값을 가진 객체의 id 값을 넣어서 해당 영역 안에서만 Popup 이미지의 이동이 가능하게 만들었다.

고급 웹 프로그래밍을 하게되면 jQuery의 다양한 기능을 많이 사용하게 되는데, 특정 객체를 화면에서 이동할 수 있게 만드는 draggable() 함수도 유용하게 사용할 수 있다. 실제 draggable() 함수에서는 마우스가 해당 객체에서 클릭된 이후에 마우스의 좌표 값을 계속 받아서 객체의 디자인 속성들('top', 'left')을 변경하는 작업을 수

행한다. 마우스의 이벤트 처리를 위한 복잡한 계산을 수행하는 JavaScript 코드가 draggable() 함수에 만들어져 있다고 이해하면 된다. 다음 장에서는 draggable() 함수에 정의된 다수의 속성들을 사용해서 다양한 기능을 하는 프로그램을 보여준다.

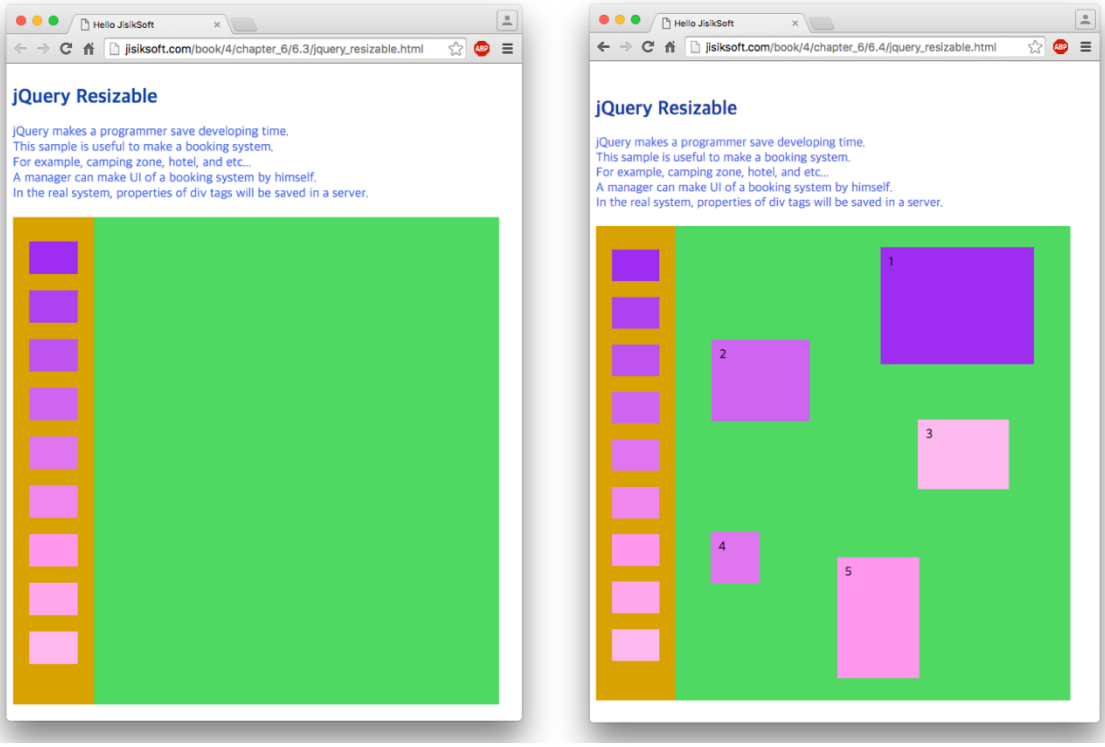
6.3 jQuery Resizable

이번 장에서 만드는 웹 프로그램은 '캠핑장'이나 '오두막'의 예약 시스템 등에 응용할 수 있는데, 캠핑장들은 서로 다른 캠핑 구역들을 가지고 있기 때문에 캠핑장 관리자가 예약시스템의 화면을 직접 구성할 수 있다.

[그림 6-5]는 왼쪽의 사각형을 사용해서 오른쪽에 다양한 구성을 할 수 있는 간단한 구조를 보여주는데, 예를 들면 캠핑장의 관리자는 오른쪽에 캠핑장 구성도를 만들고 종료하면 서버에 구성된 내용이 저장되게 만들수 있다. 이 책에서는 간단한 예제이기 때문에 도면을 구성하는 것만을 구현하였으나, 실제 예약시스템에서는 사각형의 정보들을 서버에 저장하고 일반 예약자에게는 만들어진 화면만 보여주고 예약할 수 있게 만들면 된다. 실제 서비스에서는 많은 코드들을 추가해야 하는데, 이번 장에서의 내용을 참조하면 어렵지 않게 만들수 있을 것이다. 예를 들면, 색으로 구분한 사각형에 다양한 '텐트' 이미지들을 넣으면, 캠핑장 효과를 연출할 수 있다.

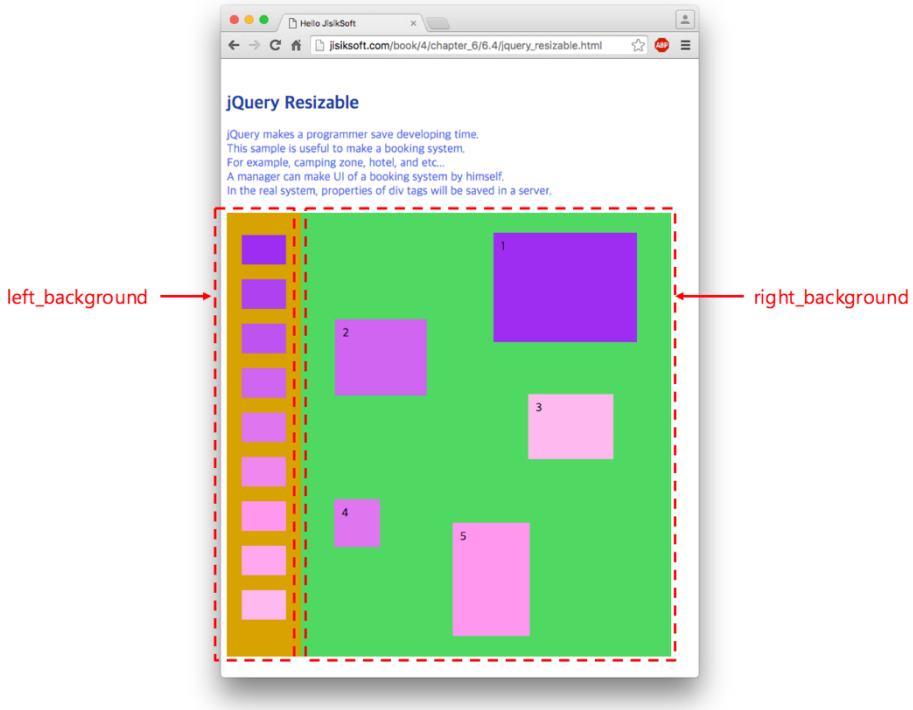
확인 사이트: http://jisiksoft.com/book/4/chapter_6/6.3/jquery_resizable.html

그림 6-5 왼쪽의 사각형을 오른쪽으로 이동한 후에 사각형 크기를 변경할 수 있는 페이지



[그림 6-6]에서 움직이는 사각형들을 두 개의 영역으로 구분한다. 왼쪽에 위치한 영역은 기본으로 사용되는 9개의 사각형을 가지고 있는 'left_background'라는 id 값을 가진 태그 객체이며, 오른쪽 영역은 'right_background'라는 id 값을 가진 태그 객체이다. 왼쪽 영역 안에 있는 사각형들은 마우스를 사용해서 이동이 가능한데, 오른쪽 영역으로 이동하면 새로운 사각형이 만들어진다. 오른쪽 영역에 새로 만들어지는 사각형들은 마우스로 이동이 가능하고 크기도 마우스를 사용해서 변경할 수 있다. 또한, 오른쪽 영역 안에 있는 사각형들은 고유한 id 값을 가지고 있으며, 이 책에서는 구현하지 않았지만 상용 프로그램으로 만든다면 id 값도 변경할 수 있게 만들어서 사용할 수 있다.

그림 6-6 기본 사각형들을 위한 왼쪽 영역과 크기를 변경할 수 있는 사각형들을 위한 오른쪽 영역



[코드 6-4]는 [그림 6-6]의 화면을 만드는 HTML 코드를 보여주는데, HTML 태그를 사용해서 왼쪽과 오른쪽의 영역을 만들어주고 JavaScript 함수를 사용해서 왼쪽 영역 안에 9개의 사각형 객체를 추가한다.

[코드 6-5]는 사각형의 영역을 초기화하고 생성되는 사각형들을 이동이 가능하고 크기를 조절할 수 있게 만드는 JavaScript 코드를 구현하였다. 이번 장에서 중요하게 생각할 것은 새로 만들어지는 사각형이 왼쪽 영역에서 만들어지는냐, 오른쪽 영역에서 만들어지느냐에 따라서 해당 사각형의 동작을 구분하게 된다. 즉, 왼쪽 영역 안

에서 만들어지는 사각형은 jQuery의 draggable() 함수만을 사용하고, 오른쪽 영역 안에서 만들어지는 사각형은 크기도 변경할 수 있게 하기 위해서 jQuery의 draggable() 함수와 resizable() 함수를 사용한다.

[코드 6-4] 사각형을 이동해서 크기를 변경하게 만드는 HTML 코드 (jquery_resizable.html)

```
##### 생략 #####

<body>
    <br><div id="subject">jQuery Resizable</div><br>
    <div id="contents">
        jQuery makes a programmer save developing time.<br>
        This sample is useful to make a booking system.<br>
        For example, camping zone, hotel, and etc...<br>
        A manager can make UI of a booking system by himself.<br>
        In the real system, properties of div tags will be saved in a server.<br>
    </div>
    <br>

    <div id="all_background">                                          //---①
        <div id="left_background" style="position: absolute; background-color: #d7a10f;">
        </div>
        <div id="right_background" style="position: absolute; background-color: #56d768;">
        </div>
    </div>
    </br>

    <script>
        init();                                                        //---②
        addLeftImages();                                               //---③
    </script>

</body>
</html>
```

① 그림이 그려지는 영역의 태그 객체로서 왼쪽 영역을 위해서 'left_background'라는 id 값을 가진 태그와 오른쪽 영역을 위해서 'right_background'라는 id 값을 가진 태그를 포함한다. 이후에 사용자에 의해서 새로 생성되는 사각형 태그 객체들은 두 개의 자식Child 태그 객체 안에 추가된다.

② 왼쪽과 오른쪽에 위치하는 두 개의 사각형의 위치와 크기를 초기화Initialize한다.

③ 왼쪽 영역에 자동으로 만들어지는 9개의 사각형을 위한 태그 객체

들을 생성한 후에 'left_background'라는 id 값을 가진 태그 객체안에 추가한다.

[코드 6-5] 사각형을 이동해서 크기를 변경할 수 있게 만드는 JavaScript 코드 (/js/jquery_resizable.js)

```
var ImageDiv = function(parentContainer, myId, myType, myRect) {                    //---①

    this.container = parentContainer;
    this.id = myId;
    this.type = myType;
    this.rect = myRect;

    this.createDiv();                                                               //---②
};

ImageDiv.prototype = {                                                              //---③

    rectLeftBackground : [ 0, 0, 100, 600 ], //[left,top,width,height]              //---④
    rectRightBackground : [ 100, 0, 500, 600 ], //[left,top,width,height]

    arrLeftImageDiv : new Array(),                                                  //---⑤
    arrRightImageDiv : new Array(),

    arrTypeColor : [ "#9d3aed", "#ad4aed", "#bd5aed", "#cd6aed", "#dd7aed",
                     "#ed8aed", "#fd9aed", "#fdaaed", "#fdbaed" ],                  //---⑥

    indexForId : 1,                                                                 //---⑦

    createDiv : function() {                                                        //---⑧

        var str = '<div id="' + this.id + '" style="position:absolute;left:'
                  + this.rect[0] + 'px;top:' + this.rect[1] + 'px;width:'
                  + this.rect[2] + 'px;height:' + this.rect[3]
                  + 'px;z-index:10;background-color:'
                  + ImageDiv.prototype.arrTypeColor[this.type]
                  + ';text-indent:10px;line-height:40px;">';
        if (this.container == 'right_background') {                                 //---⑨
            str += this.id;
        }
        str += '</div>';

        $('#' + this.container).append(str);                                        //---⑩
    },
```

```
draggableLeftImage : function() {                                    //---⑪

    var thisDiv = this;

    $('#'+thisDiv.id).draggable({                                    //---⑫
            cursor : 'move',
            containment : '#all_background',
            stop : function(event, ui) {
                var X = $(this).position().left;
                var Y = $(this).position().top;

                if (thisDiv.isMovedIntoDrawArea(X, Y)) {             //---⑬
                    var anImageDiv = new ImageDiv(      "right_background",
                        ImageDiv.prototype.indexForId,
                        thisDiv.type,
                        [ X - ImageDiv.prototype.rectRightBackground[0],
                        Y - ImageDiv.prototype.rectRightBackground[1],
                        thisDiv.rect[2],
                        thisDiv.rect[3] ]);
                    anImageDiv.draggableRightImage();
                    ImageDiv.prototype.arrRightImageDiv.push(anImageDiv);
                    ImageDiv.prototype.indexForId += 1;
                }
                thisDiv.moveToOriginPosition();
                return true;
            }
    });
},

draggableRightImage : function() {                                   //---⑭

    var thisDiv = this;

    $('#' + thisDiv.id).draggable({                                  //---⑮
            cursor : 'move',
            containment : '#right_background',
            stop : function(event, ui) {
                thisDiv.rect[0] = $(this).position().left;
                thisDiv.rect[1] = $(this).position().top;
                return true;
            }
    }).resizable({                                                   //---⑯
            stop : function(event, ui) {
                thisDiv.rect[2] = ui.size.width;
                thisDiv.rect[3] = ui.size.height;
            }
    });
},
```

```
        moveToOriginPosition : function() {                                    //---⑰

            var thisDiv = this;

            $('#' + thisDiv.id).css({
                        left : thisDiv.rect[0],
                        top : thisDiv.rect[1],
                        width : thisDiv.rect[2],
                        height : thisDiv.rect[3]
            });
        },

        isMovedIntoDrawArea : function(X, Y) {                                 //---⑱

            if ((X > this.rectRightBackground[0])
               && (X + this.rect[2] < this.rectRightBackground[0] + this.rectRightBackground[2])
               && (Y > this.rectRightBackground[1])
               && (Y + this.rect[3] < this.rectRightBackground[1] + this.rectRightBackground[3])) {
                        return true;
            } else {
                        return false;
            }
        }
}

function init() {                                                              //---⑲

        $('#left_background').css({
            left : ImageDiv.prototype.rectLeftBackground[0],
            top : ImageDiv.prototype.rectLeftBackground[1],
            width : ImageDiv.prototype.rectLeftBackground[2],
            height : ImageDiv.prototype.rectLeftBackground[3]
        });

        $('#right_background').css({
            left : ImageDiv.prototype.rectRightBackground[0],
            top : ImageDiv.prototype.rectRightBackground[1],
            width : ImageDiv.prototype.rectRightBackground[2],
            height : ImageDiv.prototype.rectRightBackground[3]
        });
}

function addLeftImages() {                                                     //---⑳

        for (var i = 0; i < 9; i++) {
            var anImageDiv = new ImageDiv('left_background', 'left_' + i, i,
                                                  [ 20, 30 + i * 60, 60, 40 ]);
            anImageDiv.draggableLeftImage();
```

```
            ImageDiv.prototype.arrLeftImageDiv.push(anImageDiv);
        }
}
```

① 새로 생성되는 사각형 객체를 만들기 위해서 ImageDiv 클래스를 만들어서 사용한다. ImageDiv 클래스는 아래에서 prototype을 사용해서 함수들을 정의했으며, 클래스 객체가 생성되면 함수를 통해 입력받는 매개변수들을 가지고 해당 클래스가 메모리에 공간을 차지한다고 이해하면 된다. (클래스에 대한 자세한 설명은 1.3장의 뒷부분에서 자세히 설명했다.) 클래스 안에서 정의된 변수들은 4개이며, container 변수는 생성된 사각형을 포함하는 영역의 id 값을 갖는데 'left_backgound' 또는 'right_background' 값을 갖는다. 그리고, id 변수는 생성되는 태그 객체의 id 값을 저장하고, type 변수는 사각형의 색을 정의하고, rect 변수는 사각형의 위치와 크기 값을 저장한다. 이렇게 정의된 클래스 객체는 id 값을 가지고 브라우저에 만들어진 태그 객체에 접근할 수 있으며, 사각형이 위치한 영역을 알 수 있다. 여기서 구현하지는 않았지만, 만약 상용 프로그램에서 만들어진 객체의 설정 값들을 서버에 저장한다면 생성된 클래스 객체들의 변수 값들을 서버에 저장하면 된다.

② ImageDiv 클래스가 생성될 때 자동으로 createDiv() 함수를 실행해서 브라우저에 사각형을 만들기 위해서 태그 객체를 만들어서 추가한다. JavaScript 언어에서 클래스를 사용한다는 것은 클래스를 위한 태그 객체를 만들어서 브라우저에 추가해서 관리해야 하는 경우에 많이 사용한다.

③ ImageDiv 클래스에서 공통적으로 사용하는 변수와 함수들을 prototype 안에서 정의하는데, prototype 안에서 정의된 내용들은 클래스가 생성될 때 중복해서 만들어지지 않고 생성된 모든 클래스 객체들이 하나의 prototype 안에서 정의된 내용을 사용한다.

④ 생성되는 사각형들은 왼쪽 영역이나 오른쪽 영역 안에 포함되며, rectLeftBackgound 변수는 'left_background'라는 id 값을 가진 사각형의 위치와 크기 값을 배열로 갖고 rectRightBackgound 변수는 'right_background'라는 id 값을 가진 사각형의 위치와 크기 값을 배열로 갖는다. 이와 같이, 왼쪽 영역과 오른쪽 영역의 위치와 크기 값을 저장하기 때문에, 만들어진 사각형이 어느 위치로 옮겨졌는지를 이후에 알 수 있다.

⑤ 왼쪽 영역에 생성된 사각형들은 arrLeftImageDiv 배열에 저장되고, 오른쪽 영역에 생성된 사각형들은 arrRightImageDiv 배열에 저장된다.

⑥ 왼쪽 영역에 자동으로 생성되는 9개의 사각형들은 다양한 색을 갖게 되는데,

arrTypeColor 배열에 저장된 9개의 색을 차례대로 갖게 된다. 이와 같이, 색 값들을 배열을 사용해서 관리하게 되면, 이후에 색 값을 변경하는 것이 간단하기 때문에 프로그램이 만들어진 이후에 유지보수가 쉽다.

⑦ 오른쪽 영역에 만들어지는 ImageDiv 클래스 객체는 id 값을 갖는데, id 값에 번호를 부여하기 위해서 indexForId 변수를 사용하고 시작 값으로 1로 정의했다. 이후에 indexForId 변수의 값을 1씩 증가시켜서 오른쪽 영역에 만들어지는 ImageDiv 클래스 객체의 id 값을 관리한다.

⑧ JavaScript 언어에서 만들어지는 클래스 객체는 브라우저 안에서 태그 객체로도 존재해야 하는데, createDiv() 함수는 태그 객체를 문자열로 만들어서 브라우저에 추가한다.

⑨ 생성되는 ImageDiv 클래스가 오른쪽 영역에 추가되는 객체라면, id 값을 화면에 보여주기 위해서 태그 안에 해당 id 값을 추가한다.

⑩ ImageDiv 클래스가 생성될 때 부모 객체의 id 값을 container 변수에 저장하였으며, append() 함수를 사용해서 객체를 정의한 문자열을 부모 객체의 마지막에 추가한다.

⑪ 왼쪽 영역에 생성된 ImageDiv 객체를 마우스로 이동할 수 있게 만드는 함수로서 jQuery의 draggable() 함수를 사용한다. 함수 안에서 "var thisDiv = this;"와 같이 현재 객체를 의미하는 this를 thisDiv 변수에 대입하는 이유는 jQuery의 draggable() 함수 안에서 사용하는 this는 다른 객체이기 때문이다. 즉, draggableLeftImage() 함수 안에서의 this는 ImageDiv 객체를 의미하고, draggable() 함수 안에서의 this는 $('#'+thisDiv.id)로 접근한 브라우저의 태그 객체를 의미한다. 이와 같이, JavaScript 언어는 this를 사용해서 어떤 객체에 접근하는지를 이해하고 프로그래밍을 해야하는데 가장 까다로우면서도 JavaScript 언어의 묘미이기도 하다.

⑫ 왼쪽 영역의 사각형을 마우스로 움직이기 위해서 사용한 jQuery의 draggable() 함수는 매개변수를 사용해서 다양한 속성값들을 받는다. 'cursor' 속성은 해당 객체가 움직일 때 마우스의 포인트를 움직이는 모양으로 변경하고, 'containment' 속성은 해당 id 값을 가진 객체 안에서만 움직일 수 있으며, 'stop' 속성은 객체의 이동이 종료되면 실행되는 함수를 정의했다. 'stop' 함수는 객체가 움직이다가 마우스의 왼쪽 버튼이 눌러지지 않았을 때 실행되는 'mouseup' 이벤트가 실행될 때 동작하는 코드를 가지고 있다.

⑬ 'stop' 함수는 객체가 움직임을 멈추었을 때의 좌표(x, y)를 X와 Y 변수에 저장한

후, 만약 오른쪽 영역으로 이동했다면 오른쪽 영역에 새로운 ImageDiv 객체를 생성한다. 'isMovedIntoDrawArea(X, Y)' 함수는 왼쪽 영역의 객체가 오른쪽 영역으로 이동했을 때 true를 반환하는데, 이때 'new ImageDiv()'를 실행해서 오른쪽 영역에 새로운 ImageDiv 클래스 객체를 생성한다. 객체가 생성된 이후에는 draggableRightImage() 함수를 실행해서 해당 객체를 오른쪽 영역에서 이동과 크기 변경이 가능하게 만들었다. 객체의 생성과 속성을 변경한 이후에 해당 객체를 arrRightImageDiv 배열에 push() 함수를 사용해서 추가하는데, 여기서는 해당 객체에 다시 접근할 필요가 없지만 배열에 넣어서 클래스 객체를 관리하면 이후에 해당 객체의 접근이 가능하다. 'stop' 함수의 마지막에는 moveToOriginPosition() 함수를 실행해서 이동한 객체를 왼쪽 영역의 원래 위치로 이동한다.

⑭ 오른쪽 영역에서 생성된 ImageDiv 클래스 객체를 이동과 크기 변경이 가능하게 만드는 함수로서, jQuery의 draggable()과 resizable() 함수를 사용한다.

⑮ 오른쪽 영역에서 생성된 ImageDiv 클래스 객체를 이동하는 함수로서 'containment' 속성을 사용해서 오른쪽 영역 안에서만 이동이 가능하고, 'stop' 함수를 사용해서 위치가 변경된 태그 객체의 위치를 ImageDiv 클래스 객체의 변수에 저장한다. "thisDiv.rect[0] = $(this).position().left;"에서 this 객체는 태그 객체를 의미하고, thisDiv 객체는 해당 ImageDiv 객체를 가리킨다.

⑯ 오른쪽 영역에서 생성된 ImageDiv 클래스 객체의 크기 변경을 가능하게 하는 함수로서, 마우스로 크기를 변경한 이후에 stop 함수가 실행된다. 'stop' 함수에서 매개변수 'ui'는 태그 객체의 변경된 크기 값을 저장하고 있으며, 'ui.size.width'는 변경된 가로 길이를 저장하고 'ui.size.height'는 변경된 세로 길이를 저장한다. 변경된 크기 값은 thisDiv 변수가 가리키는 ImageDiv 클래스 객체의 rect 변수에 저장된다.

⑰ 왼쪽 영역에 위치한 사각형들은 이동 후, 원래의 자리로 옮겨지는데 moveToOriginPosition() 함수는 해당 ImageDiv 객체의 id 변수에 저장된 값을 사용해서 해당 태그 객체의 디자인 속성을 변경한다.

⑱ 왼쪽 영역 안의 사각형이 오른쪽 영역으로 이동했는지를 확인하는 함수로서, 이동한 좌표 (X, Y)가 오른쪽 영역 안에 위치하면 true를 반환하고 그렇지 않으면 false를 반환한다.

⑲ 브라우저에서 자동으로 실행되는 함수로서, 왼쪽 영역과 오른쪽 영역의 위치와 크기의 디자인을 변경한다. 변경하는 위치와 크기는 ImageDiv의 prototype에 정의

된 값을 사용한다.

⑳ 왼쪽 영역의 9개의 사각형을 자동으로 만들어서 추가하는 함수이며, for loop을 사용해서 9번 반복해서 ImageDiv 클래스를 생성하고 draggableLeftImage() 함수를 실행해서 이동이 가능하게 만든다.

이번 장에서의 코드는 다소 복잡해 보이지만, 고급 웹 프로그래밍에서 JavaScript 언어로 클래스를 만들어서 사용하는 아주 좋은 예제이다. JavaScript에서 생성하는 클래스는 프로그래밍에서의 클래스이고, 브라우저에는 클래스와 함께 동작하는 태그 객체가 존재한다. 난이도가 있는 웹 프로그래밍은 JavaScript에서 생성한 클래스 객체를 가지고, 브라우저의 태그 객체를 얼마나 효과적으로 다루냐에 따라서 실시간으로 동작하는 동적 프로그램을 만든다. 현재 대부분의 사이트에서는 JavaScript 언어의 클래스를 사용하지 않고 개발이 이루어지지만, 앞으로는 브라우저에서 동작하는 다양한 게임들이 만들어지기 때문에 JavaScript의 클래스를 많이 사용하게 될 것이다.

6.4 Slide Bar

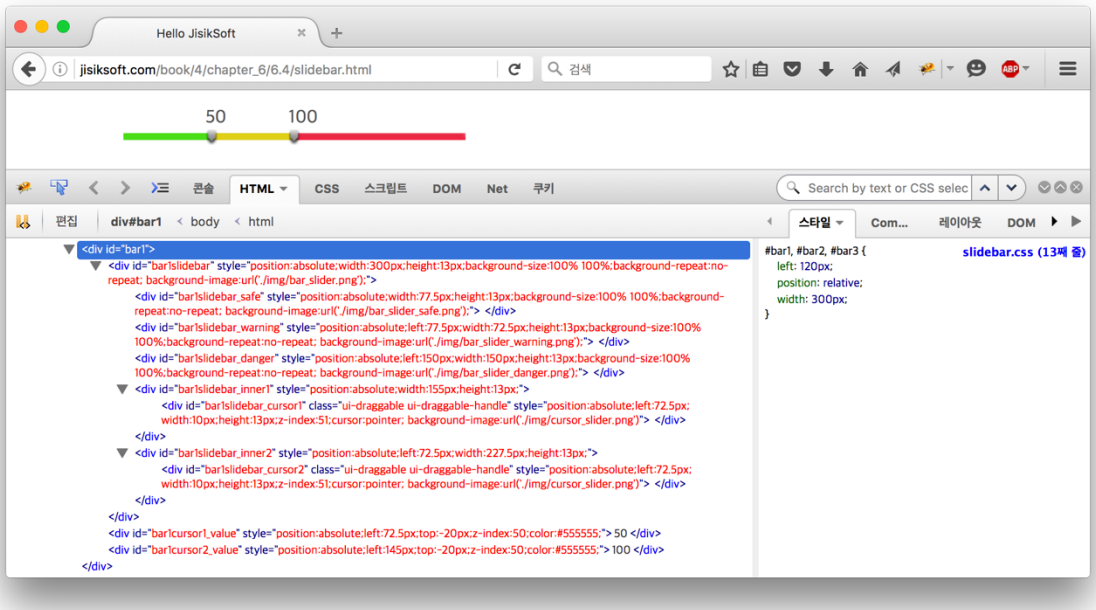
이번 장에서 만드는 'Slide Bar'는 다수의 태그들이 모여서 하나의 'Slide Bar'를 만드는 과정을 보여준다. 마우스로 'Slide Bar'의 커서Cursor를 이동시키면, 'Slide Bar'를 구성하는 모든 태그들이 실시간으로 디자인을 변경하는 과정이 반복되는 것을 이해할 수 있다. 만약 필자의 두 번째 책인 '주식 분석 프로그램 만들기'의 프로그램을 실행시켜본 독자라면 컴퓨터가 얼마나 많은 연산을 짧은 시간에 수행하는지 체감했을 것이므로, 여기서 구현하는 'Slide Bar'가 실시간으로 많은 연산과 디자인 변경 작업을 수행하지만 컴퓨터에게는 크게 무리가 되지 않다는 것을 이해할 수 있다. [그림 6-7]은 이번 장에서 구현하는 'Slide Bar'를 보여주는데, 세 개의 'Slide Bar'들은 서로 다른 수치 값을 갖는다.

확인 사이트: http://jisiksoft.com/book/4/chapter_6/6.4slidebar.html

그림 6-7 서로 다른 값을 보여주는 세 개의 'Slide Bar'

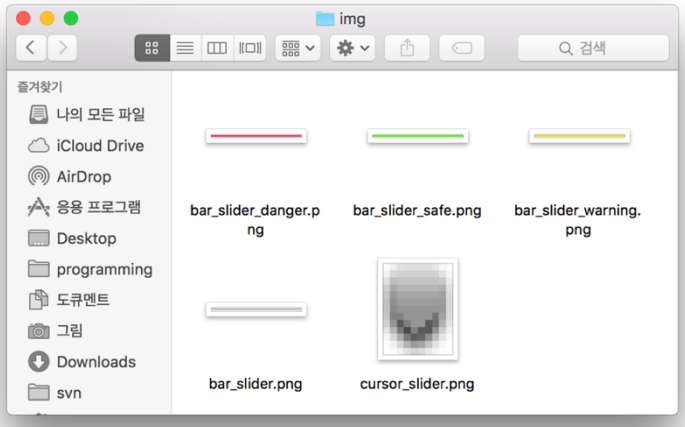
[그림 6-8]은 Firefox 브라우저에서 제공하는 Firebug라는 개발툴을 사용해서 하나의 'Slide Bar'의 코드를 확인한 결과를 보여준다. Firebug는 필자가 사용해 본 웹 프로그래밍 개발툴 중에서 가장 편리하다고 여기는 툴이며, Firebug에 대한 자세한 내용은 이 책의 부록에서 설명하였다. [그림 6-8]의 윗부분에 나오는 하나의 'Slide Bar'를 구성하는 코드는 아랫부분에 나와있는데, 이미지는 간단한 'Slide Bar'이지만 10개의 태그들이 모여서 하나의 'Slide Bar'를 만들었다. 하나의 'Slide Bar'에서 움직이는 것은 두 개의 커서이며, 이전 장의 내용과 같이 마우스로 커서를 움직이게 하기 위해서 jQuery의 draggable() 함수를 사용하면 된다. 즉, 'Slide Bar'에서 마우스에 의해 발생되는 이벤트를 처리하는 것은 draggable() 함수이며, 이 함수 안에서 마우스로 커서를 움직일 때마다 'Slide Bar'를 구성하고 있는 태그들의 디자인과 내용을 변경한다. 마우스가 커서Cursor를 한 픽셀Pixel이라도 움직일 때마다 draggable() 함수 안에서는 마우스의 위치 값을 가지고 JavaScript 코드의 연산을 수행한다.

그림 6-8 하나의 'Slide Bar'를 만드는 코드를 'Firebug'로 확인한 결과



[그림 6-9]는 'Slide Bar'를 구성하는 '/img/' 폴더안의 이미지들을 보여주는데, 보기 좋은 'Slide Bar'를 만들기 위해서 이미지들을 사용하였다. 'Slide Bar'를 구성하는 몇 개의 태그들의 바탕색을 조정해서 구현할 수도 있지만, 보기에 좋지 않았고 더 복잡한 코드를 만들수 있기 때문에 이미지들을 사용하였다.

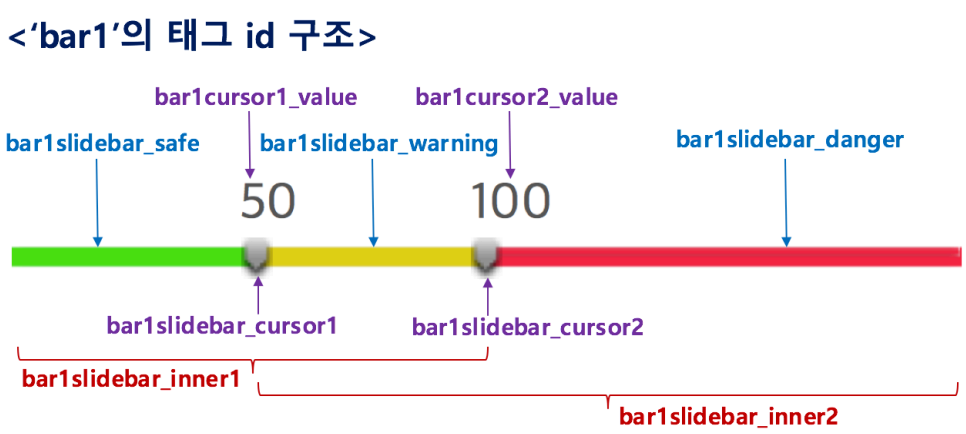
그림 6-9 'Slide Bar'를 구성하는 이미지



[그림 6-10]은 'Slide Bar'를 구성하고 있는 태그들의 id 값의 구조를 보여주는데, 'Slide Bar'의 서로 다른 이미지와 값들은 모두가 하나의 태그를 사용해서 만들었으며 태그 객체에 접근하기 위해서 고유한 id 값을 갖는다. 커서를 움직이기 위해서 사용하는 jQuery의 draggable() 함수에서 'containment' 속성을 사용해서 커서가 움직이는 범위를 설정하는데, 'bar1slidebar_cursor1'이라는 id

값을 가진 커서는 'bar1slidebar_inner1'이라는 id를 가진 사각형의 범위 안에서만 움직일 수 있게 만들었다. 이와 마찬가지로 'bar1slidebar_cursor2'라는 id 값을 가진 커서는 'bar1slidebar_inner2'라는 사각형의 범위 안에서만 움직인다. 세 가지 색상들(녹색, 노랑, 빨강)은 세 개의 태그들로 구성되어 있으며, 커서가 움직일 때마다 'Slide Bar'를 구성하고 있는 모든 태그들의 위치와 크기가 변경된다. 커서 위에는 커서가 위치한 곳의 숫자 값을 가지는데, 전체 크기에서 커서가 위치한 곳의 값을 계산하고 'bar1cursor1_value'와 'bar1cursor2_value'라는 id 값을 가진 태그들에 대입한다. 이 책에서는 사용하지 않았지만, 'Slide Bar'를 사용하는 웹 프로그램을 만든다면 커서의 위치 값들을 프로그램에서 사용하면 된다.

그림 6-10 'Slide Bar'를 구성하는 태그들은 고유한 id 를 가지고 제어된다.



[코드 6-6]은 세 개의 'Slide Bar'를 만드는 HTML 코드이며, 'Slide Bar'의 위치에 태그들을 만들어서 makeSlidebarTwoCursor() 함수를 실행시켜 자동으로 'Silde Bar'를 만든다. JavaScript에서는 이와 같이 하나의 함수를 실행시켜서 특정 기능을 하는 도구를 만드는 방법을 많이 사용한다. JavaScript 언어에서 클래스 객체를 생성하고 클래스 함수를 사용하는 방법은 클래스 개념을 이해해야 하지만, 하나의 함수로 모든 기능을 수행하게 만들어서 함수 별로 관리하는 기법도 많이 사용한다.

[코드 6-6] 'Slide Bar'를 만드는 HTML 코드 (slidebar.html)

```
##### 생략 #####

<body>
    </br><div id="subject">Slide Bar</div></br>
    <div id="contents">This coe can be applied for other slide bars you want.</div>
    </br></br></br></br>
    <div id="bar1"></div>                                          //---①
    </br></br></br>
```

```
    <div id="bar2"></div>
    </br></br></br>
    <div id="bar3"></div>
    </br></br></br>

    <script>
        makeSlidebarTwoCursor(50, 100, 200, 'bar1');                    //---②
        makeSlidebarTwoCursor(80, 250, 400, 'bar2');
        makeSlidebarTwoCursor(100, 300, 1000, 'bar3');
    </script>

</body>
</html>
```

① 화면에 세 개의 'Slide Bar'들을 나타내기 위해서, 'bar1', 'bar2', 'bar3'라는 id 값을 가진 세 개의 태그들을 만들었다. 이후에 JavaScript 함수를 사용해서 각각의 태그 안에 새로운 태그 객체들을 만들어서 'Slide Bar'의 이미지를 나타낸다.

② makeSlidebarTwoCursor() 함수를 세 번 실행해서 세 개의 'Slide Bar'들을 만든다. 함수를 실행할 때 매개변수는 '첫 번째 커서 값', '두 번째 커서 값', 'Slide Bar의 전체 값', 그리고 'Slide Bar'를 위한 태그의 id 값을 갖는다.

[코드 6-7]은 'Slide Bar'를 만드는 makeSlidebarTwoCursor() 함수를 구현하는 JavaScript 코드이며, 함수 안에서는 'Slide Bar'를 위한 새로운 태그 객체들을 만들고 커서를 위해서 만들어진 태그 객체를 jQuery의 draggable() 함수를 사용해서 움직일 수 있게 구현하였다. JavaScript 함수에서 HTML의 태그 객체들을 만들어서 브라우저에 추가하는 방법은 동적인 웹 프로그래밍에서 많이 사용하는 기법이다. [그림 6-10]의 태그 객체들에 부여된 id 값들을 이해하면 [코드 6-7]의 코드를 쉽게 이해할 수 있다. 함수 안에서 다수의 연산을 하는 과정들은 'Slide Bar' 안에 존재하는 태그들의 위치와 크기를 위한 계산들이다.

[코드 6-7] 'Slide Bar'의 태그들을 생성하고 동작시키는 JavaScript 코드 (/js/slidebar.js)

```
makeSlidebarTwoCursor = function(cur1, cur2, max, container) {          //---①

    var idSlidebar = container + 'slidebar';

    var idSafe = container + 'slidebar_safe';                           //---②
    var idWarning = container + 'slidebar_warning';
    var idDanger = container + 'slidebar_danger';
```

```
var idInner1 = container + 'slidebar_inner1';
var idCursor1 = container + 'slidebar_cursor1';
var idInner2 = container + 'slidebar_inner2';
var idCursor2 = container + 'slidebar_cursor2';

var valCursor1 = container + 'cursor1_value';
var valCursor2 = container + 'cursor2_value';

var originX = $('#'+container).offset().left;                    //---③

var width = $('#'+container).width();                            //---④

var len = width - 10;                                            //---⑤

var pos1 = (len * cur1) / max;                                   //---⑥

var pos2 = (len * cur2) / max;

var widthInner1 = pos2 + 10;                                     //---⑦

var widthInner2 = width - pos1;
var widthSafe = pos1 + 5;
var widthWarning = pos2 - pos1;
var widthDanger = width - pos2 - 5;

var str = '<div id="' + idSlidebar + '" style="position:absolute;width:' + width
          + 'px;height:13px;background-size:100% 100%;background-repeat:no-repeat; \
            background-image:url(\'./img/bar_slider.png\');"> \
          <div id="' + idSafe + '" style="position:absolute;width:' + widthSafe
          + 'px;height:13px;background-size:100% 100%;background-repeat:no-repeat; \
          background-image:url(\'./img/bar_slider_safe.png\');"></div> \
        <div id="' + idWarning + '" style="position:absolute;left:' + (pos1 + 5) + 'px;width:'
          + widthWarning + 'px;height:13px;background-size:100% 100%; \
          background-repeat:no-repeat; \
          background-image:url(\'./img/bar_slider_warning.png\');"></div> \
          <div id="' + idDanger + '" style="position:absolute;left:' + (pos2 + 5) + 'px;width:'
          + widthDanger + 'px;height:13px;background-size:100% 100%; \
          background-repeat:no-repeat; \
          background-image:url(\'./img/bar_slider_danger.png\');"></div> \
          <div id="' + idInner1 + '" style="position:absolute;width:' + widthInner1
          + 'px;height:13px;"> \
                  <div id="' + idCursor1 + '" style="position:absolute;left:' + pos1
                  + 'px;width:10px;height:13px;z-index:51;cursor:pointer; \
                  background-image:url(\'./img/cursor_slider.png\')"></div> \
          </div> \
          <div id="' + idInner2 + '" style="position:absolute;left:' + pos1 + 'px;width:'
          + widthInner2 + 'px;height:13px;"> \
                  <div id="' + idCursor2 + '" style="position:absolute;left:' + widthWarning
                  + 'px;width:10px;height:13px;z-index:51;cursor:pointer; \
                  background-image:url(\'./img/cursor_slider.png\')"></div> \
          </div> \
        </div>'
          + '<div id="' + valCursor1 + '" style="position:absolute;left:' + pos1
```

```
            + 'px;top:-20px;z-index:50;color:#555555;">' + cur1 + '</div>'
            + '<div id="' + valCursor2 + '" style="position:absolute;left:' + pos2
            + 'px;top:-20px;z-index:50;color:#555555;">' + cur2 + '</div>';      //---⑧

$('#'+container).html(str);

var slideAreaSafe = $('#' + idSafe);                                            //---⑨
var slideAreaWarning = $('#' + idWarning);
var slideAreaDanger = $('#' + idDanger);

var slideInnerFirst = $('#' + idInner1);
var slideInnerSecond = $('#' + idInner2);
var slidecursorFirst = $('#' + idCursor1);
var slidecursorSecond = $('#' + idCursor2);

var valueCursor1 = $('#' + valCursor1);
var valueCursor2 = $('#' + valCursor2);

slidecursorFirst.draggable({                                                    //---⑩

    containment : '#' + idInner1,

    drag : function(ev, ui) {                                                   //---⑪
            var z1 = parseInt(slidecursorFirst.css("z-index"), 10);
            var z2 = parseInt(slidecursorSecond.css("z-index"), 10);
            if (z1 <= z2) { slidecursorFirst.css("z-index", z2 + 1); }          //---⑫

            var posX1 = ui.offset.left - originX;                               //---⑬
            var posX2 = (len - posX1) - (len - (slidecursorSecond.offset().left - originX));

            slideInnerSecond.css({
                    left : posX1,
                    width : len - posX1 + 10
            });
            slidecursorSecond.css({
                    left : posX2
            });
            slideAreaSafe.css({
                    width : posX1 + 5
            });
            slideAreaWarning.css({
                    left : posX1 + 5,
                    width : slideInnerFirst.width() - posX1 - 5
            });

            var aValue = Math.round(((posX1) * max) / len);                     //---⑭
            valueCursor1.css({
```

```
                    left : posX1
                });
                valueCursor1.html(aValue);
        }
    });

    slidecursorSecond.draggable({                                    //---⑮

        containment : '#' + idInner2,

        drag : function(ev, ui) {                                    //---⑯
                var z1 = parseInt(slidecursorFirst.css("z-index"), 10);
                var z2 = parseInt(slidecursorSecond.css("z-index"), 10);
                if (z1 >= z2) { slidecursorSecond.css("z-index", z1 + 1); }   //---⑰

                var posX1 = slidecursorFirst.offset().left - originX;        //---⑱
                var posX2 = ui.offset.left - originX;

                slideInnerFirst.css({
                    width : posX2 + 10
                });
                slideAreaWarning.css({
                    width : posX2 - posX1
                });
                slideAreaDanger.css({
                    left : posX2 + 5,
                    width : len - posX2 + 5
                });
                var aValue = Math.round((posX2 * max) / len);        //---⑲
                valueCursor2.css({
                    left : posX2
                });
                valueCursor2.html(aValue);
        }
    });
}
```

① 'Slide Bar'를 만드는 함수로서 4개의 매개변수를 받는데, 'cur1'은 첫 번째 커서의 값이고, 'cur2'는 두 번째 커서의 값이고, 'max'는 'Slide Bar'의 최고 값이며, 'container'는 'Slide Bar'를 만드는 태그의 id 값이다.

② [그림 6-10]의 'Slide Bar'를 이루는 태그 객체들의 id 값을 만들어서 변수에 넣는다. 이와 같이 코드를 하는 이유는 가독성을 높이기 위해서이며, 'Slide Bar'는 정해진 규칙에 따라서 태그들의 id 값을 정해진다.

③ originX 변수는 'Slide Bar'의 X-축의 위치 값을 갖는데, 마우스로 커서를 이동할

때 발생하는 이벤트에서 받는 위치 값은 브라우저 전체에서의 위치 값이기 때문에 'Slide Bar'의 위치 값을 알고 있어야 'Slide Bar' 안에서의 값들을 계산할 수 있다. $('#'+container)는 'Slide Bar'가 만들어지는 태그 객체를 가져오며, jQuery의 'offset().left'를 사용해서 left 속성의 값을 가져온다.

④ width 변수는 'Slide Bar'의 픽셀Pixel단위의 가로길이를 저장하는 변수이며, makeSlidebarTwoCursor() 함수의 매개변수로 받은 'Slide Bar'의 커서의 값들을 가지고 'Slide Bar'의 화면의 비율에 맞추어서 커서의 위치를 계산하기 위해서 사용한다. 마우스로 커서를 움직일 때 변경되는 태그들의 크기는 'Slide Bar'의 가로길이를 가지고 계산한다.

⑤ 커서 이미지의 가로길이는 10px이기 때문에 'Slide Bar'의 안에 구성된 태그들의 길이 계산을 위해서 사용하는 len 변수는 'Slide Bar'의 전체 길이(width)에서 커서의 길이 10을 뺀 값을 저장한다.

⑥ pos1 변수는 첫 번째 커서의 'Slide Bar'에서의 위치값을 갖는데, 수학의 '비율 계산'을 사용해서 'Slide Bar'의 전체 값 max, 첫 번째 커서의 값 cur1, 'Slide Bar'의 화면에서의 길이 len 변수의 값들을 가지고 계산한다. (수학의 '비율 계산'은 필자의 두 번째 책 "주식 분석 프로그램 만들기"의 p50 [그림 2-14]에 설명되었다.) pos2 변수는 두 번째 커서의 'Slide Bar'에서의 위치 값을 갖으며, pos1 변수와 같은 방법으로 계산했다.

⑦ widthInner1 변수는 첫 번째 커서가 움직일수 있는 범위의 가로길이를 저장하고 있으며, jQuery의 draggable() 함수를 사용해서 커서를 이동할 때 커서는 widthInner1의 길이 안에서만 움직일 수 있다. 마찬가지로, widthInner2 변수는 두 번째 커서가 움직일 수 있는 범위의 가로길이 값을 갖는다.

⑧ 'Slide Bar'를 구성하는 모든 태그 객체들을 문자열로 만들었다. 태그 객체들의 id 값과 위치와 크기는 이전에 정해진 값을 가지고 구성하였다. JavaScript 언어에서 태그 객체들을 문자열로 만들어서 브라우저에 추가하는 방법은 웹 프로그래밍에서 많이 사용하는 기법이다.

⑨ 커서가 움직일 때 태그 객체들의 위치와 크기가 변경되는데, 'Slide Bar'의 모든 태그 객체들을 변수에 저장해서 이후에 변수를 사용해서 접근이 가능하게 만들었다. 이와 같이 jQuery를 사용해서 객체를 가져와서 변수를 사용해서 관리하는 방법은 많이 사용한다.

⑩ 첫 번째 커서를 draggable() 함수를 사용해서 이동이 가능하게 만들었으며,

'containment' 속성을 사용해서 특정 영역 안에서만 움직일 수 있다.

⑪ jQuery의 draggable() 함수에서 'drag' 속성은 마우스로 해당 객체를 이동할 때 계속적으로 실행되는 함수이다. 즉, 객체가 이동할 때 'drag(ev, ui)' 함수가 계속 실행된다고 이해하면 되는데, ui 매개변수는 이동하는 객체의 위치 값을 저장하고 있다. 'Slide Bar'에서 커서가 이동할 때, 약 0.1초마다 drag() 함수가 실행된다고 이해하면 된다.

⑫ 'Slide Bar'에는 두 개의 커서가 존재하는데, 하나의 커서가 움직여서 다른 커서와 겹쳐지면 이동한 커서의 'z-index' 값을 좀더 크게 해서 이후에 선택되는 커서는 마지막으로 움직인 커서가 되어야 한다. 그래서, 현재의 코드가 실행되는 것은 첫 번째 커서가 움직인 것이기 때문에 첫 번째 커서의 'z-index' 값을 가지고 있는 z1의 값이 두 번째 커서의 z2 값보다 항상 큰 값을 갖게 한다.

⑬ 첫 번째 커서가 움직일 때, 첫 번째 커서와 두 번째 커서의 위치 값들을 posX1과 posX2 변수에 저장한다. 'ui.offset.left'는 첫 번째 커서의 브라우저에서의 위치 값이기 때문에 브라우저에서 'Slide Bar'의 위치 값을 빼면, posX1 변수는 'Slide Bar'에서의 첫 번째 커서의 위치 값을 가진다. 두 번째 커서의 위치 값을 가지는 posX2 변수는 'Slide Bar' 안에서의 위치 값이 아닌 두 번째 커서가 이동할 수 있는 범위 안에서의 위치 값을 갖는다. 커서가 움직일 때마다 'Slide Bar'의 세 가지 색의 범위가 조정되는 것은 커서들의 위치 값을 가지고 'Slide Bar'를 구성하고 있는 태그 객체들의 위치와 크기를 변경한다.

⑭ 첫 번째 커서가 이동했기 때문에, 'Slide Bar'에서 첫 번째 커서가 위치한 곳에서의 값을 계산해서 aValue 변수에 저장하고 화면에 결과값을 출력한다. 커서가 움직일 때마다 커서 위의 값이 변경되는 것은 이러한 작업을 계속 진행했기 때문이다.

⑮ 두 번째 커서를 draggable() 함수를 사용해서 이동이 가능하게 만들었으며, 'containment' 속성을 사용해서 특정 영역 안에서만 움직이고 'Slide Bar'를 구성하는 태그 객체들의 위치와 크기를 실시간으로 변경한다.

⑯ 두 번째 커서가 이동할 때마다 실행되는 drag() 함수로서 첫 번째 커서가 이동할 때와 비슷한 동작을 수행한다. 마우스로 커서를 이동하는 동안에 0.1초 정도의 간격으로 반복적으로 발생하는 함수라고 이해하면 된다. 함수의 매개변수인 ui 변수에는 커서의 위치 값이 저장되어 있으며, 함수 안에서 이 값을 가지고 태그 객체들의 위치와 크기를 계산하고 변경한다.

⑰ 두 번째 커서가 이동했기 때문에, 두 번째 커서의 'z-index' 값을 저장하고 있는

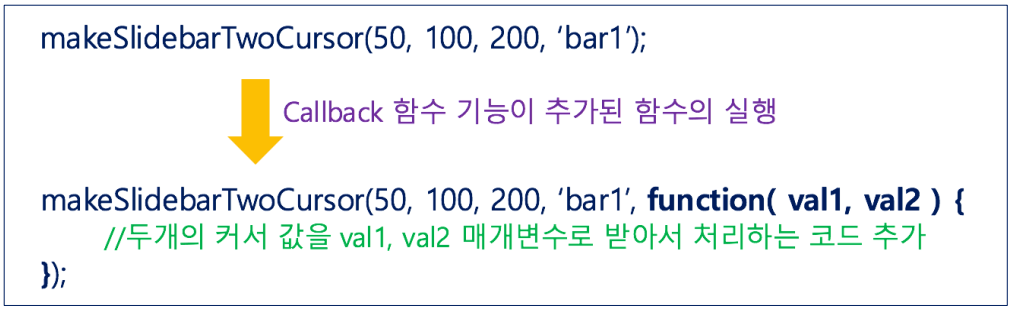
z2의 값은 첫 번째 커서의 z1보다 항상 커야한다. 두 개의 커서가 겹쳐진 상태에서는 마우스로 이동할 수 있는 커서는 두 번째 커서가 되어야 하기 때문에 이와 같이 'z-index' 값을 사용해서 화면의 앞쪽에 위치시킨다.

⑱ 'Slide Bar' 안에서 커서들의 위치를 알기 위해서 posX1과 posX2 변수를 사용했으며, posX1 변수는 첫번재 커서의 위치값을 저장하고 posX2 변수는 두 번째 커서의 위치 값을 저장한다.

⑲ 두 번째 커서가 이동했기 때문에, 'Slide Bar'에서 두 번째 커서가 위치한 곳에서의 값을 계산해서 aValue 변수에 저장하고 화면에 결과값을 출력한다.

이 책에서는 간단히 만들기 위해서 브라우저에서 동작만 하는 'Slide Bar'를 만들었지만, [그림 6-11]과 같이 1.5장의 뒷부분에서 설명한 'Callback' 함수를 사용해서 커서가 이동했을 때마다 함수를 사용하는 사람이 추가적인 코드를 구현할 수 있게 만들면 좋다. JavaScript 언어를 잘 사용하는 프로그래머는 Callback 함수를 많이 사용하며, 특정 기능을 하는 함수를 만들고 Callback 기능을 추가해서 다양한 프로그램을 구현한다. Callback 함수를 사용하였다면 makeSlidebarTwoCursor() 함수 안에서 커서가 이동을 한 이후에 커서의 위치값을 매개변수에 넣어서 Callback 함수를 실행하게 구현한다.

그림 6-11 'Slide Bar'를 만드는 함수에 Callback 기능을 추가했을 때 처리하는 방법



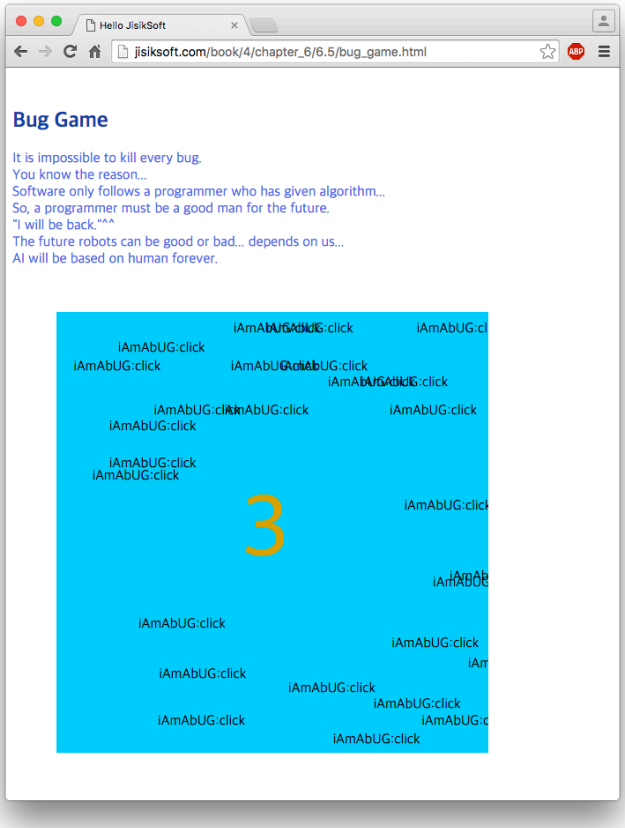
이번 장에서 설명한 'Slide Bar'와 같은 코딩 기법은 고급 웹 프로그래밍에서 아주 많이 사용한다. 일반적인 홈페이지들이 정적인 웹 프로그래밍이라면 브라우저에서 동작하는 도구들을 사용하는 웹 페이지들은 동적인 웹 프로그래밍에 해당한다. 'Slide Bar'를 만드는 JavaScript 코드를 이해하였다면 인터넷에서 보게되는 다양한 도구들을 직접 만들어보는 것도 재미있는 주제가 될수 있다.

6.5 Bug Game

이번 장에서는 간단한 'Bug' 게임을 만드는데 [그림 6-12]와 같이 'iAmAbUG:click'이라는 글자들이 계속 만들어지고 글자를 클릭하면 화면에서 사라지는 게임이다. 화면 가운데에는 클릭해서 없어진 글자들의 갯수를 카운트하는 숫자가 있다. 이 게임의 원리는 4초마다 'Bug'를 만드는 5개의 태그 객체들을 새로 추가하고, 하나의 'Bug'를 클릭하면 해당 태그의 'display' 속성을 'none'으로 설정해서 화면에서 보이지 않게 처리한다.

확인 사이트: http://jisiksoft.com/book/4/chapter_6/6.5/bug_game.html

그림 6-12 Bug Game 화면



[그림 6-13]은 'Bug Game'에서 'Bug'를 만들기 위해서 생성되는 태그 객체의 내용을 보여주는데, 모든 태그는 고유한 id 값을 가지고 있고 클래스 값을 사용해서 모든 태그 객체에 접근할 수 있다. 태그에는 name 값으로 숫자가 입력되는데 name 값에 1이 저장되어 있으면 클릭 한번에 해당 객체가 화면에서 사라지고,

2가 저장되면 두 번 클릭해야 화면에서 사라지게 된다. 즉, 클릭을 몇 번을 해야 하는지를 정하는 값을 name 값에 저장했으며, onclick 이벤트가 발생할 때마다 hitBug() 함수를 실행해서 name 값을 처리하게 된다. 'hitBug()' 함수는 실행할 때마다 해당 태그 객체의 name 값을 1씩 감소시키며, name 값이 0이면 "display:none;"으로 화면에서 해당 객체를 사라지게 한다. 'Bug Game'은 'Bug' 태그 객체를 계속해서 생성하고, 사용자가 클릭한 객체를 화면에서 사라지게 만드는 작업을 계속 수행한다.

그림 6-13 Bug 를 만드는 태그 객체 내용

Bug 태그 객체

```
<div id="bug1" class="iamabug" name="1" onclick="hitBug(this.id)">
        iAmAbUG:click
</div>
```

- 'onclick' 이벤트에 의해서 실행되는 hitBug() 함수는 name 값에서 1을 뺀다.
- 태그의 name 값이 0이면 해당 태그를 화면에서 사라지게 만든다.

[코드 6-8]과 [코드 6-9]는 'Bug Game'을 만드는 HTML과 JavaScript 코드를 보여주는데, 'Bug'를 만드는 5개의 객체들을 4초마다 주기적으로 계속 생성하고 0.5초마다 모든 버그 객체들을 가운데로 조금씩 이동하는 과정을 수행한다. 그리고, 사용자가 'Bug' 객체를 클릭하면 해당 객체를 화면에서 사라지게 한다.

[코드 6-8] 'Bug Game'을 만들기 위한 HTML 코드 (bug_game.html)

```
##### 생략 #####

<body>
    <br><div id="subject">Bug Game</div><br>
    <div id="contents">It is impossible to kill every bug.<br>
        You know the reason...<br>
        Software only follows a programmer who has given algorithm...<br>
        So, a programmer must be a good man for the future.</br>
        "I will be back."^^<br>
        The future robots can be good or bad... depends on us...<br>
        AI will be based on human forever.<br>
    </div>
    <div id="game">
        <div id="count">0</div>                                    //---①
```

```
        <div id="bug"></div>                                    //---②
        <div id="color"></div>                                  //---③
        <div class="border" id="border_top"></div>              //---④
        <div class="border" id="border_left"></div>
        <div class="border" id="border_right"></div>
        <div class="border" id="border_bottom"></div>
    </div>

    <script>
        $(window).bind("mousedown", function() {                //---⑤
                return false;
        });
        makeBugs();                                             //---⑥
        moveBugs();                                             //---⑦
    </script>

</body>
</html>
```

① 클릭해서 사라진 'Bug' 객체들을 카운트한 숫자를 화면의 가운데에 보여주기 위한 태그이다.

② 4초마다 5개씩 생성되는 'Bug' 객체들은 bug라는 id 값을 가진 객체에 추가된다.

③ 화면의 바탕색을 파란색으로 만들기 위해서 사용된 태그이다.

④ 게임의 테두리 영역을 위해서 4개의 태그들을 만들었으며, 게임의 가장자리에서 생성되는 'Bug'들을 가리는 역할을 한다. 'Bug Game'의 화면 주변에는 일정 영역을 가진 태그들이 있는데, 'z-index' 속성값을 높게 해서 화면의 테두리를 가리는 효과를 갖는다.

⑤ 화면의 글자들을 마우스의 드래그Drag로 블럭처리하는 것을 막기 위해서 mousedown 이벤트 발생 시 아무런 동작을 하지 않게 처리하였다. 게임을 하는 동작은 마우스를 클릭했을 때 발생하는 click 이벤트이기 때문에 게임을 진행하는데는 아무런 문제가 없다.

⑥ 'Bug'를 4초마다 5개씩 생성하는 함수를 실행한다.

⑦ 화면에 있는 모든 'Bug' 객체들을 0.5초마다 이동시키는 함수를 실행한다.

[코드 6-9] 움직이는 'Bug'들을 만들고 Click 하면 사라지게 하는 JavaScript 코드 (/js/bug_game.js)

```
var index = 0;                                                  //---①
var cntKilled = 0;                                              //---②
```

```
function makeBugs() {                                                    //---③

    var noClick = 1;                                                     //---④

    var WIDTH = $('#bug').width();
    var HEIGHT = $('#bug').height();

    var str = '';

    for (var i = 0; i < 5; i++) {                                        //---⑤

        var direction = Math.floor(Math.random() * 4);                   //---⑥
        var randomX = Math.random() * WIDTH;                             //---⑦
        var randomY = Math.random() * HEIGHT;
        var myId = 'bug' + index;                                        //---⑧

        str += '<div class="iamabug" id="' + myId + '" name="' + noClick + '" '   //---⑨
                + 'onclick="hitBug(this.id)" style="position:absolute;';

        if (direction == 0) {
                str += 'left:' + randomX + 'px;top:-15px;';
        } else if (direction == 1) {
                str += 'left:-30px;top:' + randomY + 'px;';
        } else if (direction == 2) {
                str += 'left:' + randomX + 'px;top:' + HEIGHT + 'px;';
        } else {
                str += 'left:' + WIDTH + 'px;top:' + randomY + 'px;';
        }
        str += 'cursor:pointer;">iAmAbUG:click</div>';

        index += 1;
    }

    $('#bug').append(str);

    setTimeout(function() {                                              //---⑩
        makeBugs();
    }, 4000);
}

function moveBugs() {                                                    //---⑪

    var originX = $('#bug').offset().left;
    var originY = $('#bug').offset().top;
    var centerX = $('#bug').width() / 2;
```

```
        var centerY = $('#bug').height() / 2;
        var curPosX, curPosY;

        $('.iamabug').each(function() {                                    //---⑫

            curPosX = $(this).offset().left - originX;
            curPosY = $(this).offset().top - originY;

            $(this).css({                                                  //---⑬
                    left : (curPosX * 40 + centerX) / 41,
                    top : (curPosY * 40 + centerY) / 41
            });
        });

        setTimeout(function() {                                            //---⑭
            moveBugs();
        }, 500);
}

function hitBug(myId) {                                                    //---⑮

        var myObject = $('#'+myId);                                        //---⑯

        var no = myObject.attr('name');                                    //---⑰
        no -= 1;
        myObject.attr('name', no);

        if (no == 0) {                                                     //---⑱
            myObject.css('display', 'none');
            cntKilled += 1;
            $('#count').empty().html(cntKilled);
        }
}
```

① 'Bug'를 만들기 위해서 생성하는 태그의 id 값에 번호를 부여하기 위해서 사용되는 index 변수는 0부터 시작해서 'Bug'를 만들 때마다 1씩 증가한다.

② 화면의 가운데에 나타나는 숫자는 클릭해서 사라진 'Bug'들의 갯수를 나타내며, cntKilled 변수에 값을 저장한다.

③ 'Bug'를 생성하는 함수로서 4초마다 5개의 Bug들을 생성하고, 생성된 Bug들은 테두리에 위치한다.

④ noClick 변수는 몇 번 클릭을 해야 Bug를 지울 수 있는지를 나타내는 값으로, 여기서는 1로 설정했기 때문에 한 번 클릭하면 Bug가 사라진다. 만약 noClick 변수의 값이 3이라면 하나의 Bug를 지우기 위해서 세 번을 클릭해야 한다. 이와 같이

클릭수를 쉽게 변경할 수 있으면, 게임을 만들 때 난이도 조절이 용이하다.

⑤ 5개의 Bug들을 만들기 위해서 for loop을 5번 실행한다. 'for loop' 안에서는 하나의 태그를 문자열로 만들며 태그의 위치 값은 JavaScript의 Math.random() 함수로 임의의 값을 만들어서 사용한다.

⑥ Bug들은 4개의 테두리에서 임의의 위치에서 생성되는데, direction 변수는 생성되는 Bug의 위치를 결정한다. JavaScript 언어에서 제공하는 Math.random() 함수는 (0,1) 사이의 실수 값을 반환하고 Math.floor() 함수는 입력되는 값보다 작은 정수를 반환하는데, "Math.floor(Math.random() * 4);" 연산은 4개의 값 {0, 1, 2, 3} 중에서 하나의 값을 반환한다. 만약 direction 값이 0이면 생성되는 Bug의 위치는 위쪽이고, 1이면 왼쪽, 2이면 아래쪽, 3이면 오른쪽에서 Bug의 위치가 정해진다.

⑦ 생성되는 Bug의 위치를 임의의 수로 정하기 위해서 Math.random() 함수를 사용했으며, randomX 변수는 X-축의 위치 값을 가지고 randomY 변수는 Y-축의 위치 값을 가진다.

⑧ Bug를 만들기 위해 생성되는 태그는 고유의 id 값은 가지며 index 변수를 사용해서 만든다. 'Bug Game'에서 하나의 Bug를 클릭하면 id 값으로 해당 객체에 접근해서 화면에서 사라지게 한다.

⑨ Bug를 위해서 생성되는 태그 객체를 문자열로 만들어서 str 변수에 저장한다. 태그의 위치는 direction 변수 값에 따라서 생성되는 방향이 정해진다.

⑩ 4초마다 5개의 Bug들을 생성하기 위해서 setTimeout() 함수를 사용해서 makeBug() 함수를 4초(4000ms) 뒤에 다시 실행한다.

⑪ 화면의 모든 Bug들을 0.5초마다 화면의 중앙으로 조금씩 이동하게 만드는 함수로서, 모든 Bug들의 태그 객체의 위치를 변경한다.

⑫ jQuery에서 클래스 값으로 객체를 가져오면 배열 형식으로 저장되며, 배열에 순차적으로 접근하기 위해서 많이 사용하는 each() 함수를 사용해서 배열 안의 객체에 하나씩 접근한다. 'jQuery'의 each() 함수 안에서 사용하는 this 객체는 순차적으로 접근한 객체를 의미한다.

⑬ this는 하나의 Bug 객체를 의미하며, 해당 객체의 위치를 변경해서 Bug가 가운데로 조금씩 움직이게 만든다. 위치의 변경되는 값은 게임 화면의 가운데로 조금씩 움직이게 계산되었다.

⑭ 0.5초마다 모든 Bug들을 움직이기 위해서 setTimeout() 함수를 사용해서 moveBugs() 함수를 0.5초(500ms) 뒤에 다시 실행한다.

⑮ 하나의 Bug를 클릭하면 실행되는 함수로서 해당 Bug 객체의 id 값을 매개변수로 받는다.

⑯ 함수의 매개변수로 받은 id 값으로 클릭된 객체를 가져와서 myObject 변수에 저장한다. 이와 같이 jQuery로 객체를 가져와서 변수를 사용해서 처리하는 방법은 웹 프로그래밍에서 많이 사용한다.

⑰ 태그의 name 값은 Bug를 없애기 위한 클릭 수를 갖고 있으며, 해당 객체의 Bug를 클릭했기 때문에 name 값을 가져와서 1을 감소시키고 다시 name 값에 저장한다.

⑱ 태그의 name 값은 클릭할 때마다 1씩 감소하는데, 만약 값이 0이라면 'display' 속성을 사용해서 해당 태그를 화면에서 보이지 않게 하고 지워진 Bug의 갯수를 1 증가시킨다.

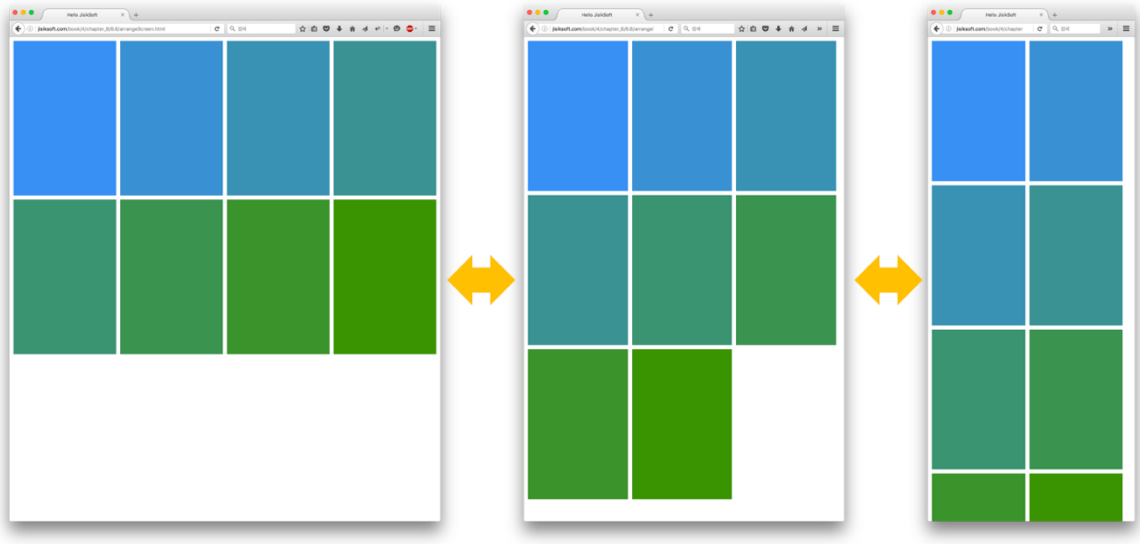
브라우저에서 동작하는 HTML, CSS, JavaScript의 내용들을 가지고 간단히 만들 수 있는 'Bug Game'을 구현하였는데, 화면을 구성하는 것은 HTML 태그들이며 디자인 속성값들을 변경해서 화면에 변화를 주는 간단한 게임이다. 인터넷 세상은 인터넷 서비스를 사용할 수 있는 브라우저가 가장 중요한 도구이기 때문에 앞으로는 브라우저에서 실행되는 많은 게임들이 만들어질 것이다. 이전에는 Flash라는 프로그램을 사용해서 많은 게임들이 만들어졌지만, 점차적으로 JavaScript로 구현되는 게임들이 늘어날 것이며 프로그래머들에게 많은 코딩작업이 요구될 수 있다. 웹 프로그래밍은 이 책의 'Chapter 1'에서 설명한 기본적인 개념을 이해하고 있으면 다양한 프로그램들을 좀더 효율적으로 만들수 있을 것이다. 최근에는 Flash를 사용해서 웹 페이지를 만들지 않는 추세인데, Flash를 사용하기 위해서는 브라우저에 추가적인 프로그램을 설치해야하고 Flash로 만든 데이터는 이미지 데이터에 기반하기 때문에 용량이 상당히 크기 때문이다. 우리나라와 같이 인터넷 속도가 빠른 곳은 큰 용량의 데이터를 주고받는데 큰 불편을 느끼지 못하지만, 많은 나라들의 인터넷 속도가 아직은 느리기 때문에 Flash로 만든 화면을 실행하는데에 시간이 다소 소요된다. 또한, 많은 사용자에게 서비스를 제공하는 시스템이 Flash로 구성되면, 동시에 다수의 사용자가 웹 페이지에 접속할 때 원활한 서비스가 안 될 수 있다. 인터넷을 하다가 브라우저가 하얀 페이지로 오랫동안 멈춰있다면 해당 서비스에 과도한 트래픽이 발생해서 서비스가 지연되는 상황이라고 판단할 수 있다.

6.6 화면 구성 변경

한때 브라우저의 크기가 변경되면 화면을 구성하고 있는 내용의 변경이 자동적으로 진행되는 홈페이지들이 있었다. JavaScript 코드에서 브라우저의 크기를 계산해서 일정 범위에서 화면을 구성하고 있는 내용들의 크기와 위치를 변경하는 동작을 진행하는데, [그림 6-14]는 브라우저의 크기에 따라서 다수의 사격형 크기와 위치가 변경되는 화면을 보여준다. 여기서는 사각형의 바탕색을 다르게 해서 구분하였는데, 실제 홈페이지에서는 사각형을 만드는 태그에 이미지를 삽입해서 화려하게 만들 수 있다. 또한 onclick 이벤트를 사용해서 하나의 사각형을 클릭하면 다른 페이지로 이동하는 동작 등을 추가해서 홈페이지를 다양하게 구성할 수 있다.

확인 사이트: http://jisiksoft.com/book/4/chapter_6/6.6/arrangeScreen.html

그림 6-14 브라우저의 크기에 따라 화면의 배열이 변경된다.



브라우저의 크기가 변경될 때마다 JavaScript 코드를 실행해서 화면의 구성을 변경하는 방법은 크게 두 가지를 유의하면 된다. 첫 번째는 브라우저의 크기가 변경될 때마다 자동으로 resize 이벤트가 실행된다는 것, 두 번째는 브라우저의 크기는 window.innerWidth 변수에 저장되어 있다는 것이다. 브라우저에서 홈페이지가 보여지는 곳은 창Window이다. 이 window는 홈페이지를 보여주는 객체를 의미한다. 'jQuery'를 사용해서 $(window).width()와 같이 window 객체의 가로길이Width를 가져올 수도 있지만, JavaScript에서 window를 사용하면 window 객체를 의미하기 때문

에 이미 정의된 'window.innerWidth'를 사용해서 가로길이를 가져온다.

[코드 6-10]은 사각형을 만드는 태그를 사용해서 8개의 사각형을 브라우저에 만들고, 브라우저의 크기가 변경될 때마다 발생하는 'resize' 이벤트를 처리하는 코드를 보여준다.

[코드 6-10] 브라우저 크기를 변경하면 사각형의 크기와 위치를 변경하는 코드 (arrangeScreen.html)

```
<!DOCTYPE html>

<head>
<title>Hello JisikSoft</title>
<meta charset="utf-8"></meta>
<link rel="stylesheet" href="./css/arrangeScreen.css"></link>
<script src="./js/jquery-2.1.3.min.js"></script>
</head>

<body>
    <div class="cell" id="cell1"></div>                                    //---①
    <div class="cell" id="cell2"></div>
    <div class="cell" id="cell3"></div>
    <div class="cell" id="cell4"></div>
    <div class="cell" id="cell5"></div>
    <div class="cell" id="cell6"></div>
    <div class="cell" id="cell7"></div>
    <div class="cell" id="cell8"></div>

    <script>

        var timerResize;                                                   //---②

        $(window).bind("resize", function() {                              //---③
                clearTimeout( timerResize );
                timerResize = setTimeout(function(){
                        arrangeCell();
                }, 300);
        });

        arrangeCell();                                                     //---④

        function arrangeCell() {                                           //---⑤

                var width = window.innerWidth;                             //---⑥
                var count;
```

```
                if (width < 600) {                                          //---⑦
                        count = 2;
                } else if (width < 1000) {
                        count = 3;
                } else {
                        count = 4;
                }
                var widthCell = (width - 50) / count;                       //---⑧

                $('.cell').each( function(index){                           //---⑨
                        $(this).animate({
                                'left': (widthCell + 10) * (index % count) + 10,
                                'top': ((widthCell * 1.5) + 10) * Math.floor(index / count) + 10,
                                'width': widthCell,
                                'height': (widthCell * 1.5)
                        }, 1000);
                });
        }
    </script>
</body>
</html>
```

---

① 8개의 태그들을 사용해서 화면에 8개의 사각형을 만들었으며, 각각의 사각형은 고유한 id 값을 가진다.

② 브라우저의 크기가 변경될 때마다 resize 이벤트가 발생해서 화면의 구성을 변경하는 JavaScript 코드를 실행하는데, setTimeout() 함수를 사용해서 0.3초 뒤에 화면을 변경한다. 'timeResize' 변수는 setTimeout() 함수가 실행될 때 반환하는 값을 저장한다.

③ jQuery의 bind() 함수를 사용해서 window 객체에서 resize 이벤트가 발생하면 실행되는 함수를 정의했다. 가장 먼저 실행되는 clearTimeout() 함수는 이전에 실행된 setTimeout() 함수의 실행을 취소하는데, 0.3초 뒤에 실행되는 arrangeCell() 함수보다 resize 이벤트가 먼저 발생하면 arrangeCell() 함수의 실행을 취소한다. 브라우저의 크기가 변경될 때마다 반복적으로 resize 이벤트가 발생하는데, 브라우저의 크기를 변경하는 중간에 발생한 resize 이벤트에 의해서 arrangCell() 함수가 실행되는 것을 막기 위해서 clearTimeout() 함수를 사용했다. 즉, 브라우저의 크기가 변경된 이후에 발생하는 마지막 resize 이벤트에서 실행되는 setTimeout() 안의 arrangeCell() 함수만 실행한다. setTimeout() 함수와 clearTimeout() 함수를 사용해서 브라우저의 크기가 완전히 변경된 이후에 arrangeCell() 함수를 한번만 실행하는 방법은 웹 프로그래밍에서 자주 사용하는 방법이다.

④ 브라우저에서 모든 코드를 다운받으면 arrangCell() 함수를 자동으로 실행해서 현재 브라우저 크기에 맞는 화면 구성을 진행한다.

⑤ arrangeCell() 함수는 브라우저에서 처음에 한번 자동실행하고, 이후에 브라우저의 크기가 변경될 때마다 실행한다. 브라우저의 크기에 따라서 화면 안의 8개의 사각형의 크기와 위치를 변경한다.

⑥ window 객체의 innerWidth 변수는 브라우저의 가로길이를 저장하고 있으며, 이 값을 가져와서 width 변수에 저장한다.

⑦ count 변수는 화면의 한 줄에 몇 개의 사각형이 위치해야 하는지를 저장하는 변수이다. 화면의 가로길이가 600 픽셀 이하이면 한 줄에 2개의 사각형을 보여주고, 1000 픽셀 이하이면 3개, 1000 픽셀 이상이면 4개의 사각형이 한 줄에 위치한다.

⑧ 한 줄에 위치 해야하는 갯수를 저장하고 있는 count 변수를 기반으로 화면 안의 사각형Cell의 가로길이를 계산해서 widthCell 변수에 저장한다.

⑨ 8개의 사각형을 만드는 태그들은 클래스 값으로 'cell'을 갖는데, $('.cell')과 같이 크래스 값을 사용해서 jQuery로 객체에 접근하면 8개의 객체들이 선택되어 배열Array에 저장된다. jQuery의 each() 함수를 사용해서 배열에 저장된 객체들에 하나씩 접근하는데, 함수의 매개변수인 index는 배열의 위치 값을 갖고 함수 안의 this는 차례대로 선택된 하나의 객체를 의미한다. 예를 들면, each() 함수는 function(index) 함수를 8번 실행하는데, 첫 번째 실행할 때는 index 값이 0을 갖고 this는 첫 번째 태그 객체를 가리킨다. 또한 두 번째 실행할 때의 index 값은 1을 의미하고 this는 두 번째 태그 객체를 가리킨다. 모든 태그 객체들은 animate() 함수를 실행해서 크기와 위치를 변경하는데, index와 count 변수의 값에 의해서 위치가 계산된다.

웹 프로그래밍을 포함한 많은 프로그래밍에서 수학적인 계산을 필요로 하는데, 다른 곳에서 만들어진 코드를 재사용하는 것보다는 자신만의 새로운 코드를 만들어서 좀 더 다양한 기능을 하는 프로그램밍하는 것이 좋다. 동적인 웹 프로그래밍에서는 브라우저의 크기에 따라 화면을 구성하는 객체의 크기를 변경할 때가 많으며, 이번 장에서 구현한 방법을 응용하면 좀더 화려한 웹 프로그래밍이 가능하다. JavaScript 언어는 사용자에 의해서 발생하는 이벤트를 처리하기 위해서 사용한다는 것을 이해하면, 브라우저의 크기도 결국은 사용자에 의해서 변경되기 때문에 resize 이벤트도 사용자 이벤트로 이해할 수 있다.

chapter 7

HTML5 프로그래밍

웹 프로그래밍은 브라우저에서 동작하는 프로그램을 만든다고 이해하면 되는데, 모든 브라우저들은 다수의 운영체제에서 동작하는 어플리케이션이기 때문에 모든 종류의 브라우저가 동일하게 동작하지 않는다. 즉, 브라우저를 만드는 회사들은 특정 동작을 수행하는 브라우저로 발전시키면서 많은 사용자들이 자사의 브라우저를 사용하기를 원한다. 브라우저는 HTML, CSS, JavaScript 언어로 만들어진 코드들을 해석하고 화면에 출력하는 기능을 하는데, 모든 브라우저에서 같은 동작을 하기 위해서 표준화 작업을 진행하고 있다. 브라우저에서 해석하는 기본 언어는 HTML이며, HTML 언어에서 정의한 태그들을 기본으로 모든 브라우저가 개발되고 있다. HTML의 개념이 처음 발표된 이후로 표준화 작업은 계속되어 왔으며, 현재는 5번째 표준화 버젼이라는 개념으로 'HTML5'라고 부른다. 'HTML5'에서는 좀더 다양한 기능을 위해서 추가된 항목들이 있는데, 운영체제의 어플리케이션에서 동작하는 기능을 브라우저에서도 사용할 수 있도록 발전하고 있다. HTML5에서 추가된 내용은 화면에 그림을 그릴 수 있는 Canvas의 추가가 대표적이며, Canvas를 사용해서 입체적인 그림을 제공하는 WebGL이 추가되었다. 또한, HTML5에는 음악과 동영상을 쉽게 웹 페이지에 넣을 수 있는 태그가 추가되었고, 웹 페이지를 종료해도 원하는 데이터를 보관할 수 있는 WebStorage 기능이 있다. 브라우저는 하나의 어플리케이션이기 때문에 운영체제에서 하나의 프로세스로 동작하는데, 고급 프로그래밍에서 사용하는 Multi-Tasking 개념을 사용하기 위해서 WebWorker 기능을 만들었다. WebWorker는 현재 페이지를 관리하는 프로세스가 아닌, 운영체제의 다른 프로세스를 사용해서 프로그램을 동작시킨다고 이해하면 된다. HTML5에서 제공하는 편리한 기능 중의 하나는 'ServerSent Event'인데, 지금까지는 브라우저에서 서버에 데이터를 요청해서 받는 과정이 중요했다면 'ServerSent Event'는 서버에서 약 3초 간격으로 데이터를 계속 전송하고 브라우저는 데이터를 받았을 때 발생하는 이벤트를 사용해서 브라우저에 데이터를 업데이트한다. 'ServerSent Event'를 사용하면 서버의 모니터링이나 실시간 스포츠 내용을 사용자에게 전송할 때 편리한데, 예를 들면 사용자가 브라우저

를 보고 있을때 일정한 간격으로 서버에 데이터를 요청하는 것보다 서버의 데이터 내용이 변경되었을 때 사용자에게 전송하는 것이 다수의 사용자들에게 서비스 하는 시스템에서는 효과적이다.

현재 인터넷 상의 많은 웹 페이지들은 HTML5에서 추가된 기능들을 사용하지 않지만, 점차적으로 브라우저에서 다양한 기능을 제공하는 페이지들이 늘어날 것이다. 회사 홈페이지들은 홍보만를 위해서 만들어진 것이 많지만, 실시간으로 변경되는 정보를 제공하는 곳에서는 HTML5의 기능들을 사용해서 쉽게 웹 페이지들을 제작할 수 있다. 또한, 텍스트 위주의 웹 페이지에서 입체적으로 움직이는 그림을 그려주거나, 브라우저에서 즐길 수 있는 게임 제작에 WebGL 기능을 사용해서 다양한 효과를 줄 수 있다. 이 책에서는 HTML5에서 추가된 기능 중에 WebSocket과 WebSQL은 설명하지 않는다. WebSocket은 브라우저에서 다수의 사용자가 함께 하는 게임을 제작할 때 사용하면 좋지만, Socket에 대한 설명과 Server에서 동작하는 프로그램을 구현하기 위해서 이 책에서 다루지 않는 프로그래밍 언어를 설명해야 하기 때문에 생략한다. 또한, WebSQL은 브라우저에서 데이터베이스를 사용해서 만들어야 하는 프로그램이 없다고 필자가 생각하기 때문이다. 필자는 상용화된 소프트웨어 개발에 참여해서 브라우저에서 동작하는 모니터링 화면을 만들면서 HTML5의 다양한 기능들을 사용했는데, 이번 Chapter를 이해한 독자들도 고급 웹 프로그래밍에서 HTML5의 기능들을 효과적으로 응용해서 멋진 프로그램을 만들기 바란다.

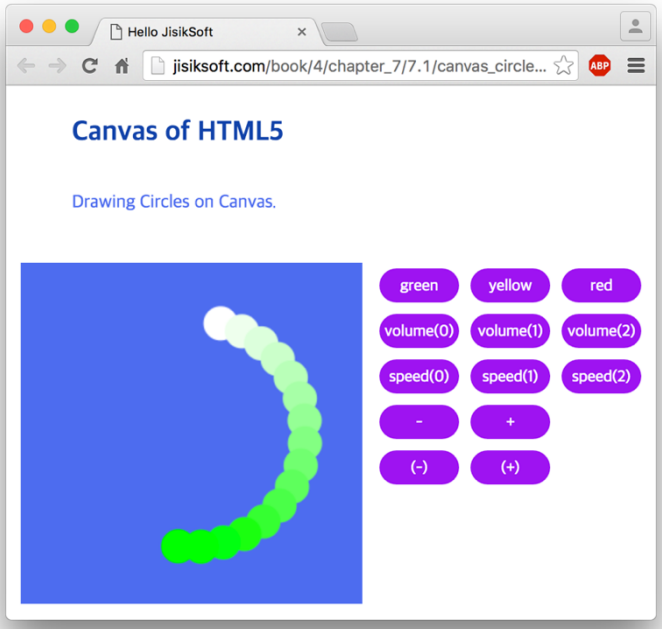
7.1 Canvas (2D Graphic)

국어사전에서 'Canvas'(캔버스)는 "유화를 그릴 때 사용하는 천"이라는 뜻으로 나온다. HTML5에서도 프로그래머가 그림을 그리는 환경을 제공하기 위해서 <canvas></canvas> 태그를 만들었다. <canvas></canvas> 태그에서 그림을 그리기 위해서 프로그래머는 붓 대신 JavaScript 언어를 사용해서 그림을 그린다. 한장의 멋진 사진을 그릴 수 있지만, 조금씩 변하는 사진을 반복적으로 계속 그리면 움직이는 애니메이션을 만들 수도 있다.

[그림 7-1]은 Canvas에서 움직이는 효과를 보여주는 화면이며, 오른쪽 버튼들을 클릭해서 다양한 변화를 줄수 있다.

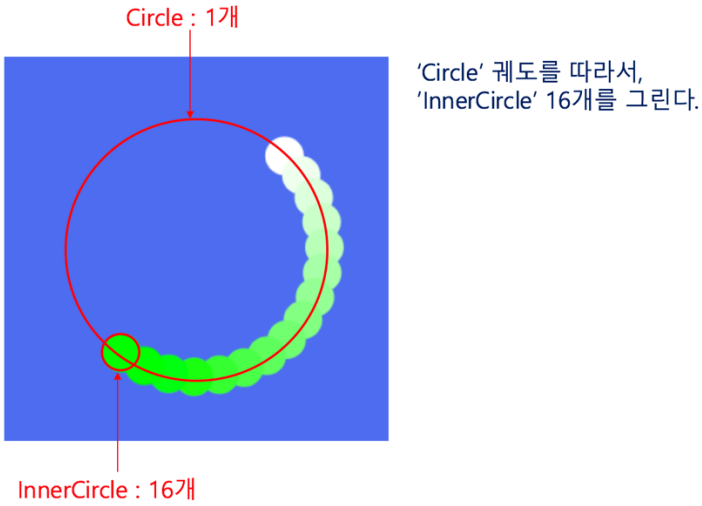
확인 사이트: http://jisiksoft.com/book/4/chapter_7/7.1/canvas_circle.html

그림 7-1 Canvas 에 그림을 그려서 움직이는 효과를 표현할 화면



[그림 7-2]는 Canvas에 그리는 그림을 수학적으로 설명하는데, 'Circle'이라는 큰 원이 존재하고 'Circle'의 궤도에 'InnerCircle'이라는 16개의 원을 그리는 과정을 보여준다. 'Circle'은 원의 궤도에 대한 위치값을 정하기 위해서 사용하며, 16개의 'InnerCircle'은 원의 중심값이 다른 동일한 크기의 원들을 그려주며 색에 변화를 주었다. Canvas에서 움직이는 원들을 그려주는 방법은 일정한 간격으로 모든 그림을 지우고 조금씩 위치를 변경해서 새로운 그림을 그리는 동작을 반복한다. 움직이는 그림을 위해서 짧은 시간 간격으로 다수의 원을 그리더라도 아주 빠른 연산을 수행하는 컴퓨터에게 큰 무리를 주는 작업은 아니다.

그림 7-2 큰 원을 중심으로 16 개의 작은 원들을 그린다.



HTML5의 Canvas 기능에서 그림을 그리는 방법은 <canvas></canvas> 태그 객체 안에 정의된 Context 객체의 함수들을 사용해서 그림을 그린다. 움직이는 효과를 주기 위해서 크기가 정해진 사각형인 <canvas></canvas> 태그 객체의 모든 내용을 지우고 16개의 원을 그리는 과정을 반복한다. [코드 7-1]은 HTML 코드에 그림을 그리기 위한 <canvas></canvas> 태그를 추가한 코드를 보여주며, [코드 7-2]는 움직이는 그림을 그리기 위한 JavaScript 코드를 보여준다. [코드 7-1]에는 다수의 버튼을 위한 태그들이 있으며 버튼이 클릭될 때마다 [코드 7-2]의 함수들을 실행해서 Canvas에 그려지는 그림의 속성들을 변경한다.

[코드 7-1] 그림을 그리기 위해서 <canvas></canvas> 태그가 추가된 HTML 코드 (canvas_circle.html)

```
<!DOCTYPE html>

<head>
	<title>Hello JisikSoft</title>
	<meta charset="utf-8"></meta>
	<link rel="stylesheet" href="./css/canvas_circle.css"></link>
	<script src="./js/jquery-2.1.3.min.js"></script>
	<script src="./js/canvas_circle.js"></script>
</head>
<body>
	<br>
	<div id='subject'>Canvas of HTML5</div>
	<br><br>
	<div id='contents'>Drawing Circles on Canvas.</div>
	<br><br>
	<div id="areaCanvas">                                                    //---①
	  <canvas id="canvas" width="300" height="300"></canvas>
	</div>
	<div id="areaButton">                                                    //---②
	  <div class="button btn1" onclick="circle.changeColor('green');">green</div>
	  <div class="button btn2" onclick="circle.changeColor('yellow');">yellow</div>
	  <div class="button btn3" onclick="circle.changeColor('red');">red</div>
	  <div class="lineEmpty""></div>
	  <div class="button btn1" onclick="circle.changeVolume(0);">volume(0)</div>
	  <div class="button btn2" onclick="circle.changeVolume(1);">volume(1)</div>
	  <div class="button btn3" onclick="circle.changeVolume(2);">volume(2)</div>
	  <div class="lineEmpty""></div>
	  <div class="button btn1" onclick="circle.changeSpeed(0);">speed(0)</div>
	  <div class="button btn2" onclick="circle.changeSpeed(1);">speed(1)</div>
	  <div class="button btn3" onclick="circle.changeSpeed(2);">speed(2)</div>
	  <div class="lineEmpty"></div>
	  <div class="button btn1" onclick="circle.decreaseCircle();">-</div>
	  <div class="button btn2" onclick="circle.increaseCircle();">+</div>
	  <div class="lineEmpty"></div>
```

```
        <div class="button btn1" onclick="circle.decreaseInnerCircle();">(-)</div>
        <div class="button btn2" onclick="circle.increaseInnerCircle();">(+)</div>
    </div>

    <script>
        $(window).bind("mousedown", function() {
                return false;
        });
        var circle = new Circle("canvas");                                  //---③
    </script>
</body>
</html>
```

① 그림을 그리기 위한 <canvas></canvas> 태그를 포함하는 'areaCanvas'라는 id 값을 가진 태그는 Canvas의 크기와 위치를 정하기 위해서 사용했다. 또한 CSS 디자인으로 Canvas의 바탕색을 여기서 정했으며, 상용 프로그램을 만들 때는 특정 이미지를 캔버스 밑그림으로 사용할 때가 많다.

② Canvas의 움직이는 그림을 그릴 때 다양한 속성값을 변경하기 위해서 사용하는 버튼들을 태그들로 만들었다.

③ Canvas에 그림을 그리는 Circle() 클래스를 생성하고 circle 변수에 넣었다. 브라우저에서 자동으로 Circle() 클래스가 생성되기 때문에 이후에 버튼을 클릭할 때 Circle() 클래스의 함수들을 실행할 수 있다.

[코드 7-2] 다양한 속성값을 가지고, 움직이는 그림을 그리는 JavaScript 코드 (/js/canvas_circle.js)

```
var InnerCircle = function() {                                              //---①
};

InnerCircle.prototype = {

    color :                                                                 //---②
[[ '#0f0','#1f1','#2f3','#3f3','#4f4','#5f5','#6f6','#7f7','#8f8','#9f9','#afa','#bfb','#cfc','#dfd','#efe','#fff' ],
[ '#ff0','#ff1','#ff2','#ff3','#ff4','#ff5','#ff6','#ff7','#ff8','#ff9','#ffa','#ffb','#ffc','#ffd','#ffe','#fff' ],
[ '#f00','#f11','#f22','#f33','#f44','#f55','#f66','#f77','#f88','#f99','#faa','#fbb','#fcc','#fdd','#fee','#fff' ]],

    radius : 15,                                                            //---③
    colorType : 0,

    posX : [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ],             //---④
    posY : [ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ],

    draw : function(context) {                                              //---⑤
```

```
            for (var i = 15; i >= 0; i--) {
                    context.fillStyle = this.color[this.colorType][i];
                    context.beginPath();
                    context.arc(this.posX[i], this.posY[i], this.radius, 0, Math.PI * 2);    //---⑥
                    context.fill();
            }
        }
}

var Circle = function(container) {                                                          //---⑦

        var context = null;                                                                 //---⑧
        var innerCircle = null;                                                             //---⑨
        var centerPoint = null;
        var widthCanvas = null;

        this.innerCircle = new InnerCircle();                                               //---⑩
        this.setCircle(container);
        this.changeSpeed(0);
};

Circle.prototype = {

        timerDraw : null,                                                                   //---⑪

        arrVolume : [ 0.07, 0.11, 0.2 ],                                                    //---⑫
        arrSpeed : [ 200, 80, 20 ],
        arrColor : [ 'green', 'yellow', 'red' ],

        radius : null,                                                                      //---⑬
        angle : 0,
        volume : 0.07,
        speed : 200,

        setCircle : function(container) {                                                   //---⑭

            this.context = document.getElementById(container).getContext('2d');             //---⑮
            this.widthCanvas = $('#' + container).width();
            this.centerPoint = this.widthCanvas / 2;
            this.radius = this.widthCanvas / 3;

            var startX = this.centerPoint + Math.cos(this.angle) * this.radius;             //---⑯
            var startY = this.centerPoint + Math.sin(this.angle) * this.radius;
```

```
    for (var i = 0; i < 15; i++) {                                    //---⑰
            this.innerCircle.posX[i] = startX;
            this.innerCircle.posY[i] = startY;
    }
},

draw : function() {                                                   //---⑱

    for (var i = 15; i > 0; i--) {                                    //---⑲
            this.innerCircle.posX[i] = this.innerCircle.posX[i - 1];
            this.innerCircle.posY[i] = this.innerCircle.posY[i - 1];
    }
    this.angle += this.volume;                                        //---⑳

    this.innerCircle.posX[0] = this.centerPoint + Math.cos(this.angle) * this.radius;
    this.innerCircle.posY[0] = this.centerPoint + Math.sin(this.angle) * this.radius;

    this.context.clearRect(0, 0, this.widthCanvas, this.widthCanvas);  //---㉑
    this.innerCircle.draw(this.context);
},

changeColor : function(color) {                                       //---㉒
    var index = this.arrColor.indexOf(color);
    if ((index < 0) || (index >= this.arrColor.length)) { return false; }
    this.innerCircle.colorType = index;
},

changeVolume : function(value) {                                      //---㉓

    if (value < 0) { value = 0; }
    if (value > 2) { value = 2; }
    this.volume = this.arrVolume[value];
},

changeSpeed : function(value) {                                       //---㉔

    if (value < 0) { value = 0; }
    if (value > 2) { value = 2; }
    this.speed = this.arrSpeed[value];
    clearInterval( this.timerDraw );

    var circle = this;
    this.timerDraw = setInterval(function() {
            circle.draw();
    }, circle.speed);

},
```

```
    increaseCircle : function() {                                    //---㉕
        if (this.radius > 150) { return false; }
        this.radius += 3;
    },

    decreaseCircle : function() {                                    //---㉖
        if (this.radius < 10) { return false; }
        this.radius -= 3;
    },

    increaseInnerCircle : function() {                               //---㉗
        if (this.innerCircle.radius > 30) { return false; }
        this.innerCircle.radius += 3;
    },

    decreaseInnerCircle : function() {                               //---㉘
        if (this.innerCircle.radius < 1) { return false; }
        this.innerCircle.radius -= 3;
    }
}
```

① InnerCircle() 클래스는 화면에 16개의 작은 원들을 그린다.

② 화면의 버튼에서 3개의 색들(Green, Yellow, Red)을 선택할 수 있는데, color[0]에는 Green의 16가지 색들이 정의되었고, color[1]에 Yellow의 16가지 색들과 color[2]에 Red의 16가지 색들이 정의되었다.

③ radius 변수는 작은 원의 반지름 값을 갖는데 15 픽셀Pixel로 초기화 되었으며, 화면의 버튼에 따라서 radius 변수값이 변경된다. ColorType 변수는 color 배열의 순서를 의미하는데, colorType이 0이면 color[0]의 색을 정하고, 1이면 color[1]을 의미하고, 2이면 color[2]의 색을 정한다.

④ posX와 posY 변수는 16개 원들의 가운데 점의 위치값(x, y)을 의미하며, 순서가 i-번째 원의 중심 좌표는 (posX[i-1], posY[i-1])의 값을 갖는다.

⑤ draw() 함수는 16개의 원들을 그리며, 매개변수로 받은 context는 Canvas에 그림을 그리기 위한 객체이다. Canvas에 그림을 그리는 객체인 context의 함수들을 사용하는데, fillStyle은 그리는 색을 설정하고 beginPath()는 그리는 과정을 시작하고 arc() 함수를 사용해서 원을 그리고 fill() 함수를 사용해서 설정된 색을 원안에 채운다.

⑥ context 객체의 arc() 함수는 원을 그리는데, 원의 가운데 점의 위치를 정하기 위해서 posX[i]와 posY[i] 값을 사용했고, 원의 반지름은 radius 값으로 정해지고, 0도에서 시작해서 360도(Math.PI * 2)까지 호(arc)를 그려서 원을 만든다.

⑦ Canvas에 그림을 그리는 클래스이며, Circle 클래스가 생성된 이후에 클래스 함수들을 사용해서 움직이는 그림의 속성들을 변경한다. Circle() 생성자의 매개변수인 container는 <canvas></canvas> 태그의 id 값이고, 해당 Canvas 객체에서 그림을 그리는 Context 객체를 가져와서 그림을 그린다.

⑧ <canvas></canvas> 태그 객체에서 그림을 그리기 위해서 사용하는 Context 객체를 context 변수에 저장한다.

⑨ 16개의 원들을 그리기 위해서 생성한 InnerCircle 객체를 가리키는 변수로서, InnerCircle 클래스가 생성된 이후에 innerCircle.draw() 함수를 실행해서 16개의 원들을 그린다. centerPoint 변수는 큰원의 궤도를 계산하기 위해서 원의 가운데 위치값을 저장하는데, Canvas의 가로와 세로의 크기가 같기 때문에 원의 가운데 위치값(x,y)은 (centerPoint, centerPoint)로 정해진다. 마찬가지로, Canvas의 가로와 세로 길이가 같기 때문에 widthCanvas에 가로길이만 저장해서 Canvas의 크기를 알수있다.

⑩ Circle 클래스가 생성되면 자동적으로 InnerCircle 클래스를 생성하고 innerCircle 변수가 가리킨다. 이후에 setCircle() 함수를 사용해서 그림을 그리기 위한 준비작업을 진행하고, changeSpeed() 함수를 실행해서 일정한 간격으로 그림을 반복해서 그린다. changeSpeed() 함수 안에서 setInterval() 함수를 사용해서 일정한 간격으로 그림을 새로 그리는 draw() 함수를 실행해서 움직이는 효과를 만든다.

⑪ 반복적으로 그림을 그리기 위해서 사용하는 setInterval() 함수를 실행할 때 반환되는 값을 저장하기 위해서 timerDraw 변수를 사용한다. 회전하는 원이 빠르게 움직이는 이유는 그리는 간격을 빠르게 하는 것이며 setInterval()에서 시간을 작게 하면 빠르게 동작하는 그림이 되는데, changeSpeed() 함수에서 실행되는 setInterval()을 다시 시작하기 전에 clearInterval() 함수로 이전 실행을 종료하기 위해서 timerDraw 변수를 사용한다.

⑫ 화면에서 'Volume' 버튼을 클릭하면 원들 간의 간격이 변하는데, 원의 간격을 정하는 세 개의 값을 arrVolume 배열에 저장했다. 화면의 그림을 그리는 간격을 작게 하면 빨리 움직이는 그림을 만드는데, 그림을 그리는 간격의 세 가지 값을 arrSpeed 배열에 저장했다. 화면에서 원의 색은 세 가지이며 arrColor 배열에 저장했다.

⑬ radius 변수는 큰원을 나타내는 Circle의 반지름 값을 저장하고, angle 변수는 큰원의 궤적에서 특정 위치값을 갖기 위한 각도를 저장한다. volume 변수는 작은 원들간의 간격을 조정하기 위해서 angle 변수의 각도에 더해지는 값을 저장한다. 즉,

angle 변수의 값에 큰 값이 더해지면 작은원들간의 간격은 벌어지고, 작은 값이 더해지면 간격은 작아진다.

⑭ Circle 클래스가 생성되면 자동적으로 실행되는 함수로서, 그림을 그리기 위해서 Context 클래스 객체를 가져오고 Circle 클래스에서 사용하는 변수들의 값을 설정한다.

⑮ document.getElementById() 함수를 사용해서 <canvas></canvas> 태그 객체를 가져오고, getContext() 함수를 사용해서 해당 Canvas에서 그림을 그릴 수 있는 Context 클래스를 가져와서 context 변수에 저장한다. 또한 <canvas></canvas> 태그의 가로길이를 기준으로 widthCanvas, centerPoint, radius 변수들의 값을 정한다.

⑯ [그림 7-2]에서 큰원을 의미하는 Circle의 궤적에서 위치를 정하기 위해서 startX와 startY 변수를 사용하는데, 원의 가운데 위치(centerPoint)를 기준으로 삼각함수의 Sin과 Cos 함수를 사용해서 특정 각도(angle)와 반지름(radius)를 가지고 위치(startX, startY)를 계산한다.

⑰ 큰원을 의미하는 Circle의 시작 위치를 16개의 작은원들의 가운데 위치값으로 설정한다. 이와 같은 초기화 작업이 진행된 이후에 16개의 원들은 같은 위치에서 움직이기 시작한다.

⑱ 움직이는 효과를 주기 위해서 반복적으로 실행되는 draw() 함수는 16개 작은 원들의 위치값을 변경하고, 화면을 지우고 16개의 원들을 다시 그리는 작업을 수행한다.

⑲ 모든 원들의 위치가 차례대로 변경되기 때문에 for loop을 사용해서 posX와 posY 배열의 값들을 순차적으로 뒤로 보낸다.

⑳ 움직이는 원들 중에 가장 앞에서 움직이는 것처럼 보이는 원의 위치값은 (posX[0], posY[0])이며, 변경되는 위치를 계산하기 위해서 angle 값에 volume을 더하고 새로운 위치값을 계산해서 posX[0]과 posY[0] 변수에 저장한다.

㉑ context의 clearRect() 함수를 사용해서 Canvas의 사각형 영역을 모두 지우고 16개의 원을 다시 그리기 위해서 innerCircle.draw() 함수를 실행한다.

㉒ changeColor() 함수는 16개의 작은 원들의 색을 벼경한다. 배열의 indexOf() 함수는 해당 컬러의 배열에서의 위치값을 반환하고, index 값에 저장된 배열의 위치값을 InnerCircle 클래스의 colorType 변수에 대입한다.

㉓ 16개의 작은원들의 간격을 변경하는 changeVolume() 함수는 volume 변수의 값을 arrVolume 배열에 있는 값으로 변경하고, 이후에 그림을 그릴 때 자동으로 간격

이 조정된다.

㉔ 그림을 다시 그리는 간격을 짧게 하면 속도가 빠른 효과를 주는데, changeSpeed() 함수는 그림을 그리는 간격을 정하는 speed 변수의 값을 변경하고 setInterval() 함수는 speed 변수 값을 가지고 실행한다. setInterval() 함수를 실행하기 전에는 항상 clearInterval() 함수를 실행해서 이전에 설정된 setInterval() 함수의 실행을 취소한다. setInterval() 함수에서는 draw() 함수를 주기적으로 실행시키며, 실행 간격은 speed 변수값에 따른다.

㉕ increaseCircle() 함수는 [그림 7-2]에서 큰원을 의미하는 Circle의 반지름을 증가시킨다.

㉖ decreaseCircle() 함수는 [그림 7-2]에서 큰원을 의미하는 Circle의 반지름을 감소시킨다.

㉗ increaseInnerCircle() 함수는 [그림 7-2]에서 16개의 작은원들을 의미하는 InnerCircle의 반지름을 증가시킨다.

㉘ decreaseInnerCircle() 함수는 [그림 7-2]에서 16개의 작은원들을 의미하는 InnerCircle의 반지름을 감소시킨다.

지금까지 Canvas에서 그림을 주기적으로 그리는 방법으로 움직이는 효과를 만들었는데, 모든 동영상은 많은 사진들을 차례대로 보여주면서 움직이는 효과를 만드는 것으로 이해하면 된다. 스포츠와 같은 움직임이 많은 동영상을 시청할 때 초당 보여주는 프레임Frame 수(fps: Frame Per Second)를 중요하게 여기는데, 12-fps 동영상은 초당 12개 사진들을 화면에 차례대로 보여준다. 스포츠를 볼 때 선수의 모든 움직임이 자세히 보여지는 동영상은 1초에 많은 사진을 보여주는 것이며, 깨끗하지 않은 동영상은 초당 보여지는 프레임 수가 적다.

7.2 WebGL (3D Graphic)

오래전부터 입체적인 그림(3D: 3-Dimension)을 그리기 위해서 사용한 프로그래밍 언어는 'OpenGL'이며, '마인크래프트'나 '카운터스트라이크' 같은 '1인칭 슈팅게임'은 OpenGL 언어를 사용해서 만들었다. 3D 그래픽은 좌표로서 (x, y, z)를 사용하는데, 평명 좌표(x, y)에 z-축을 추가한 것이다. 'OpenGL'은 (x, y, z) 좌표에서 계산으로 선

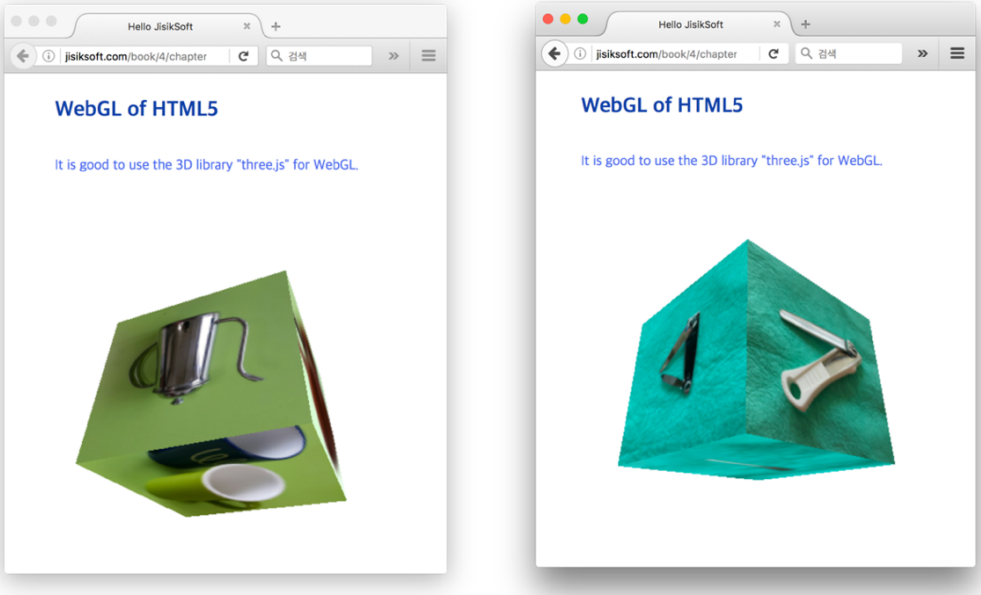
과 면을 만드는데, 코드로 입체적인 그래픽을 구현하는 'OpenGL' 언어를 잘 다루기 위해서는 수학적 지식이 많아야 한다. 이번 장에서 다루는 'WebGL'은 'OpenGL'을 브라우저에서 사용하기 위해서 변형한 것이며, 많은 부분이 'OpenGL'의 원리를 따르고 있다. WebGL을 사용해서 3D 그래픽을 하기 위해서는 이전 장의 Canvas에 입체적인 공간을 계산해서 (x, y, z) 좌표를 기준으로 만들고자 하는 대상을 그리고, 움직이는 동작을 위해서는 반복해서 계산을 진행하고 다시 그리는 작업을 진행하는 원리로 이해하면 된다. 이러한 과정들을 웹 프로그래머가 모두 구현하기에는 배워야 할 내용들이 많고, 구현하는데 시간이 상당히 소요된다. 그래서, jQuery와 같이 WebGL에서도 기본적인 항목들을 구현한 후에 웹 프로그래머에게 편리하게 사용할 수 있게 제공한 라이브러리Library들이 다수가 있는데, 현재 가장 많이 사용하는 WebGL 라이브러리에는 'three.js'(http://threejs.org)가 가장 유명하다. 인터넷의 3D 예제들의 반 이상은 'three.js' 라이브러리를 사용해서 만든 것이고, 'three.js'를 기본으로 확장된 기능들을 추가해서 새로운 함수들을 만들고 있다. 1.4장에서 라이브러리들은 함수들의 집합이라고 했는데, 'three.js'도 3D 그래픽을 만들기 위해서 다양한 기능을 하는 JavaScript 함수들을 만들어서 웹 프로그래머들이 쉽게 사용할 수 있게 제공한 것이다.

[그림 7-3]은 이번 장에서 'three.js' 라이브러리를 사용해서 구현한 '움직이는 정육면체'의 화면을 보여준다.

확인 사이트: http://jisiksoft.com/book/4/chapter_7/7.2/webgl.html

확인 사이트: http://jisiksoft.com/book/4/chapter_7/7.2/webgl2.html

그림 7-3 6 개의 사진을 보여주는 정육면체가 회전하는 화면



브라우저는 코드를 해석해서 화면에 그려주는 프로그램이며, 코드의 기본은 HTML 언어의 태그에 기반한다고 이해하면 된다. 그래서 'three.js'와 같은 라이브러리의 함수를 사용해서 화면에 그림을 그리는 것은 함수에서 태그를 자동으로 만들어서 화면에 삽입하는 원리다. 추가된 태그 객체 안에서 JavaScript 함수가 주기적으로 계산해서 그림을 조금씩 변경하고 그리는 작업이 반복되기 때문에 브라우저를 실행시키는 컴퓨터의 CPU 사용률이 조금 올라간다. [코드 7-3]은 '움직이는 정육면체'를 만들기 위해서 'cube'라는 id 값을 가진 태그를 만들었는데, [코드 7-4]에서 'three.js' 라이브러리를 사용하면 <canvas></canvas> 태그가 자동으로 추가되고 Canvas 객체 안에서 그림을 그린다. 또한 움직이기 때문에 JavaScript 함수 안에서 계속해서 계산을 하면서 조금씩 변경되는 새로운 그림을 화면에 반복적으로 그린다.

[코드 7-3] 움직이는 정육면체를 만들기 위한 HTML 코드 (webgl.html)

```
<!DOCTYPE html>

<head>
	<title>Hello JisikSoft</title>
	<meta charset="utf-8"></meta>
	<link rel="stylesheet" href="./css/webgl.css"></link>
	<script src="./js/jquery-2.1.3.min.js"></script>
	<script src="./js/three.min.js"></script>
	<script src="./js/webgl.js"></script>

</head>

<body>
	<br>
	<div id='subject'>WebGL of HTML5</div>
	<br><br>
	<div id='contents'>It is good to use the 3D library "three.js" for WebGL.</div>
	<br>
	<div id='cube'></div>                                                    //---①

    <script>

    initialize( 'cube', 500 );                                               //---②
    animate();                                                               //---③

    </script>

</body>
</html>
```

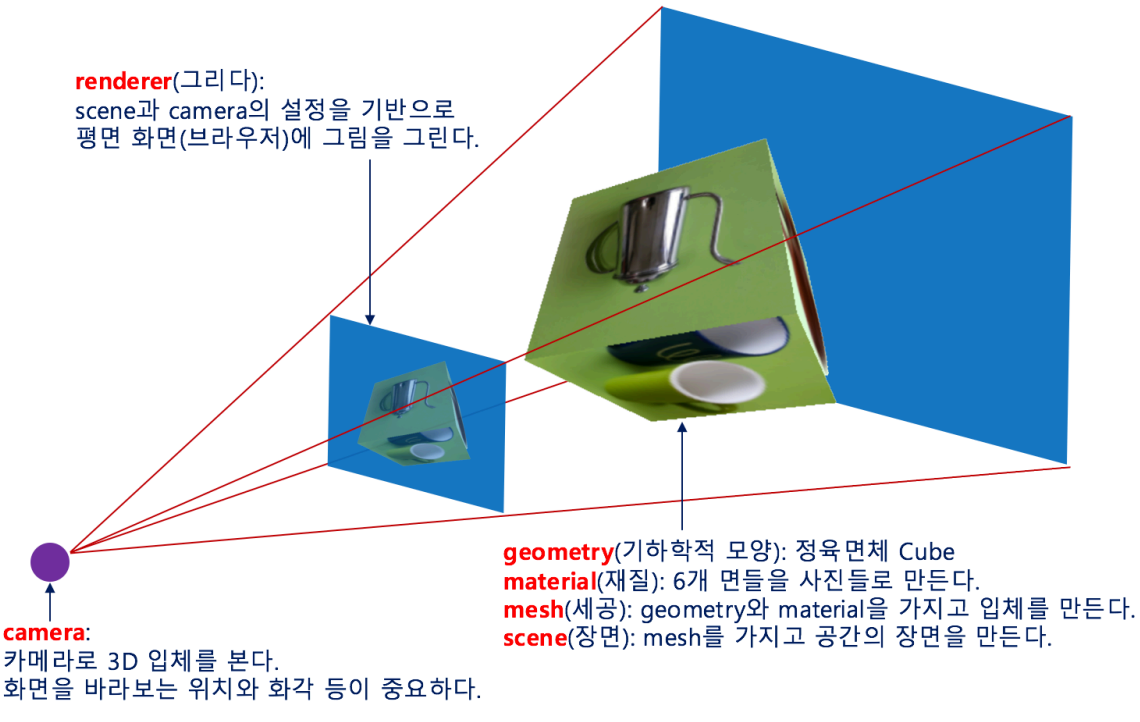
① '움직이는 정육면체'를 만들기 위해서 'cube'라는 id 값을 가진 태그를 추가했으며, 'webgl.css'에서 디자인 속성으로 크기를 가로와 세로를 동일하게 500px로 정했다.

② 'three.js' 라이브러리의 함수를 사용해서 '움직이는 정육면체'의 구조를 초기화한다. 매개변수로 전달하는 '500'은 그림을 그리는 정사각형 화면의 가로길이로서 'cube'라는 id 값을 가진 태그의 크기를 의미한다.

③ '움직이는 정육면체'의 이미지를 조금씩 변경하면서 반복해서 화면에 그린다.

'three.js' 라이브러리를 사용하면 입체적인 공간을 브라우저라는 평면의 화면에 효과적으로 표현하기 위해서 상당히 깊이있는 코드를 구현했음을 알 수 있는데, 우리가 일상에서 사진을 찍을 때 사용하는 카메라의 관점에서 함수들을 만들었다. [그림 7-4]는 'three.js' 라이브러리의 함수들을 기반으로 3D를 이해하는 그림을 보여주는데, 'camera'는 공간에서 입체를 바라보는 위치와 카메라의 화각에 따라서 사물이 평면에서 다르게 표현되는 현상을 구현했다. 공간에서의 사물을 3D로 만들기 위해서 기하학적 모양을 'geometry'로 정의하며, '움직이는 정육면체'의 표면을 만들기 위해서 'material'을 사용해서 6개의 면들을 사진으로 만들었다. 'mesh'는 'geometry'와 'material'을 사용해서 공간의 물체를 만들고, 'scene'을 사용해서 'mesh'의 구조를 공간에 표현한다. 마지막으로 'renderer'를 사용해서 'scene'을 바라보는 'camera'의 관점에서의 그림을 평면(브라우저)에 그린다. 화면에 그림을 그리는 Renderer(렌더러)는 대부분의 영상 프로그래밍에서 사용하는 용어이다.

그림 7-4 'three.js' 라이브러리를 사용해서 그림을 그리는 방법



[코드 7-4]는 'three.js' 라이브러리의 함수들을 사용해서 WebGL을 구현하는 JavaScript 코드이며, [그밀 7-4]의 과정을 'three.js'의 함수들을 차례대로 사용해서 '움직이는 정육면체'를 만든다.

[코드 7-4] 'three.js' 라이브러리의 함수를 사용해서 움직이는 정육면체를 만드는 코드 (webgl.js)

```
var camera, mesh, scene, renderer;                                    //---①

function initialize( container, size ) {                              //---②

    camera = new THREE.PerspectiveCamera( 70, size/size, 1, 1000 );   //---③

    camera.position.z = size;                                         //---④

    var geometry = new THREE.CubeGeometry( 250, 250, 250 );           //---⑤

    var material = new THREE.MeshFaceMaterial([                       //---⑥
        new THREE.MeshBasicMaterial(
                { map: THREE.ImageUtils.loadTexture( './img/1.jpg' ) } ),
        new THREE.MeshBasicMaterial(
                { map: THREE.ImageUtils.loadTexture( './img/2.jpg' ) } ),
        new THREE.MeshBasicMaterial(
                { map: THREE.ImageUtils.loadTexture( './img/3.jpg' ) } ),
        new THREE.MeshBasicMaterial(
                { map: THREE.ImageUtils.loadTexture( './img/4.jpg' ) } ),
        new THREE.MeshBasicMaterial(
                { map: THREE.ImageUtils.loadTexture( './img/5.jpg' ) } ),
        new THREE.MeshBasicMaterial(
                { map: THREE.ImageUtils.loadTexture( './img/6.jpg' ) } )
    ]);

    mesh = new THREE.Mesh( geometry, material );                      //---⑦

    scene = new THREE.Scene();                                        //---⑧
    scene.add( mesh );

    renderer = new THREE.WebGLRenderer();                             //---⑨
    renderer.setSize( size, size );
    renderer.setClearColor( 0xffffff, 1);
    $('#'+container).html( renderer.domElement );                     //---⑩
```

```
}

function animate() {                                                    //---⑪
        requestAnimationFrame( animate );
        mesh.rotation.x += 0.005;
        mesh.rotation.y += 0.01;
        renderer.render( scene, camera );
}
```

① 'three.js' 라이브러리의 함수를 사용해서 생성하는 클래스들을 저장하기 위한 변수들이다.

② 3D로 만드는 정육면체(Cube)를 초기화하는 함수로서, 매개변수로 받는 'container'는 정육면체가 만들어지는 태그 객체의 id 값이고 'size'는 태그 객체의 가로길이의 값이다.

③ 'camera'를 생성하는 PerspectiveCamera() 클래스의 첫 번째 매개변수인 70은 화면에서 수직으로 보여지는 영역을 의미하는데, 이 값이 작아지면 사물의 작은 영역을 화면에 크게 보여준다. 두 번째 매개변수는 'camera'에서 보는 영역의 가로와 세로의 비율을 정하는 값이며, 여기서는 정사각형의 값을 갖기 대문에 'size/size'(= 1)의 값으로 정했다. 나머지 매개변수들은 가까이에 있는 평면과 가장 멀리있는 평면을 의미하는데, 3D로 만들어지는 사물이 보여지는 범위를 정한다. PerspectiveCamera() 생성자를 자세히 이해하기 위해서는 매개변수들의 값들을 변경하면서 사물이 보여지는 범위를 확인하는 것이 좋다.

④ 카메라의 위치를 z-축으로 size(= 500)만큼 이동시킨다.

⑤ 정육면체(Cube)를 만들기 위해서 CubeGeometry() 생성자를 사용해서 클래스를 만든다. 함수의 매개변수는 Cube의 가로, 세로, 높이를 정하는 값이다.

⑥ 정육면체를 구성하는 'material'을 생성하기 위해서 MeshFaceMaterial() 함수를 사용했으며, MeshFaceMaterial() 생성자는 정육면체의 면들을 디자인하는 함수로서 MeshBasicMaterial() 생성자를 사용해서 6개 면을 6개의 사진으로 덮는다.

⑦ Mesh() 생성자는 위에서 만들어진 'geometry'와 'material' 클래스들을 사용해서 3D 입체 사물을 구성한다.

⑧ Scene() 생성자로 'scene' 클래스를 만들고 여기에 이미 생성된 'mesh' 클래스를 더해서 공간에서 사물이 보여지는 장면Scene을 만든다.

⑨ 화면에 그림을 그리기 위해서 WebGLRenderer() 생성자로 'renderer' 클래스를 생성하는데, setSize() 함수는 그리는 영역의 크기를 정하고 setClearColor() 함수는 배경색과 투명도를 정한다.

⑩ 'renderer' 클래스는 Canvas에 그림을 그리는데, 그림을 그리기 위해서 자동으로 생성한 <canvas></canvas> 태그 객체를 화면에서 그리기 위한 객체에 삽입한다. 여기서는 'container' 값이 'cube'이기 때문에, 'cube'라는 id 값을 가진 태그 안에 <canvas></canvas> 태그를 추가한다.

⑪ animate() 함수는 '움직이는 정사각형'을 그리기 위해서 반복적으로 실행되는 함수로서 일반적으로 JavaScript 언어에서 제공하는 setTimeout()이나 setInterval() 함수를 사용하지는 않았지만, requestAnimationFrame() 함수를 사용해서 매개변수의 'animate'를 Callback 함수로 사용해서 animate() 함수를 주기적으로 계속 실행한다. 3D로 만든 정사각형Cube을 회전하기rotation 위해서 x-축과 y-축의 값을 조금씩 증가하면서 그림을 그린다. 그림을 그리는 'renderer' 클래스의 reder() 함수는 'camera'와 'scene' 클래스를 가지고 브라우저 화면에 그림을 그린다.

라이브러리를 사용할 때는 라이브러리에서 제공하는 함수들의 사용법을 정확히 알고 사용해야 하기 때문에 이해하는 과정에서 다소 어려움을 느낄 수 있다. 하지만, 'three.js'와 같이 많은 프로그래머들이 이용하는 라이브러리들은 사용법을 설명하는 문서Document를 잘 만들었기 때문에 만들고자 하는 3D 이미지의 예제를 충분한 시간을 두고 다루다 보면 쉽게 이해할 수 있다. 3D 이미지를 만들기 위해서 처음부터 모든 것들을 구현하는 것보다는 이미 만들어진 'three.js'와 같은 라이브러리를 사용하면 웹 프로그래밍에서 많은 시간을 절약할 수 있다.

7.3 Video / Audio

HTML5에는 <video></video> 태그와 <audio></audio> 태그가 추가되어 동영상과 음악을 쉽게 웹 페이지에 삽입할 수 있다. 이번 장에서는 지금까지 만든 웹 프로그램에서 가장 간단한 코드를 보여주는데, [그림 7-5]는 동영상과 음악을 브라우저에서 보여준다.

확인 사이트: http://jisiksoft.com/book/4/chapter_7/7.3/videoAudio.html

그림 7-5 동영상과 음악을 보여주는 화면

[코드 7-5]는 <video></video> 태그와 <audio></audio> 태그가 추가된 코드이며, <source> 태그를 사용해서 동영상과 음악을 다운받을 수 있는 사이트를 연결했다. <video></video> 태그와 <audio></audio> 태그에는 다양한 속성을 부여할 수 있는데, 'autoplay'는 동영상이나 음악을 브라우저에서 자동으로 실행시키고 'loop'는 동영상이나 음악이 끝났을 때 처음부터 다시 실행시키면서 무한 반복을 실행한다. <video></video> 태그에서는 'muted' 속성을 사용해서 동영상에서 소리가 안나게 설정해서 브라우저에서는 <audio></audio> 태그의 소리만 출력한다. 'autoplay' 대신에 'controls' 속성을 사용해서 동영상이나 음악을 제어할 수 있는 영역을 만들 수도 있는데, 태그들의 속성들을 필요에 따라 적절히 사용하는 것이 좋다.

[코드 7-5] 동영상과 음악을 브라우저에 추가한 코드 (videoAudio.html)

```
<!DOCTYPE html>

<head>
<title>Hello JisikSoft</title>
<meta charset="utf-8"></meta>
</head>

<body>
	<br><br>
```

```
    <video style="width:480px;height:320px;" autoplay loop muted>                    //---①
        <source src="http://jisiksoft.com/repository/video/swimming.mp4" type="video/mp4">
    </video>

    <audio autoplay loop>                                                            //---②
        <source src="http://jisiksoft.com/repository/audio/서형무_약속을 지키는 남자.mp3"
                type="audio/mpeg">
    </audio>
    <br><br>

    <div style="position:relative;left:25px;font-size:20px;color:navy;line-height:30px;">
        2015년, 수영대회에 첫 출전한 지식소프트.<br>
        물론, 참여자는 지식소프트 부장 한명이다.<br>
        시작이 꼴찌이니, 앞으로 더 못할 수는 없다. 다행이다.^^<br>
        <br>
        고향 친구 한명이 가수다.<br>
        저작권료 안내고 노래 하나를 사용할 수 있어서 고맙다.<br>
        동영상은 보기 안 좋아도 노래가 좋으니 봐주세요.<br>
        음악은 '서형무'의 "약속을 지키는 남자"입니다.<br>
        많이 사랑해 주세요.^^<br>
    </div>

</body>
</html>
```

① 'style' 디자인 속성을 사용해서 브라우저에서 보여지는 동영상의 크기를 정했으며, 페이지가 브라우저에서 열릴 때 자동으로 실행되고(autoplay) 무한반복을 진행하며(loop) 소리는 나오지 않는다.(muted) <source> 태그를 사용해서 동영상의 위치와 파일형식(type)을 설정했다.

② <audio></audio> 태그에서 'controls' 속성을 사용하면 화면에 음악을 제어하는 영역이 표시되기 때문에 위치를 정해주어야 하지만, 'autoplay'를 사용해서 음악이 자동으로 나오게 설정했기 때문에 브라우저에서의 위치에 특별히 신경쓰지 않아도 된다. 'loop' 속성을 사용해서 음악은 반복해서 실행되고, <source> 태그를 사용해서 음악의 위치와 파일형식(type)을 설정했다.

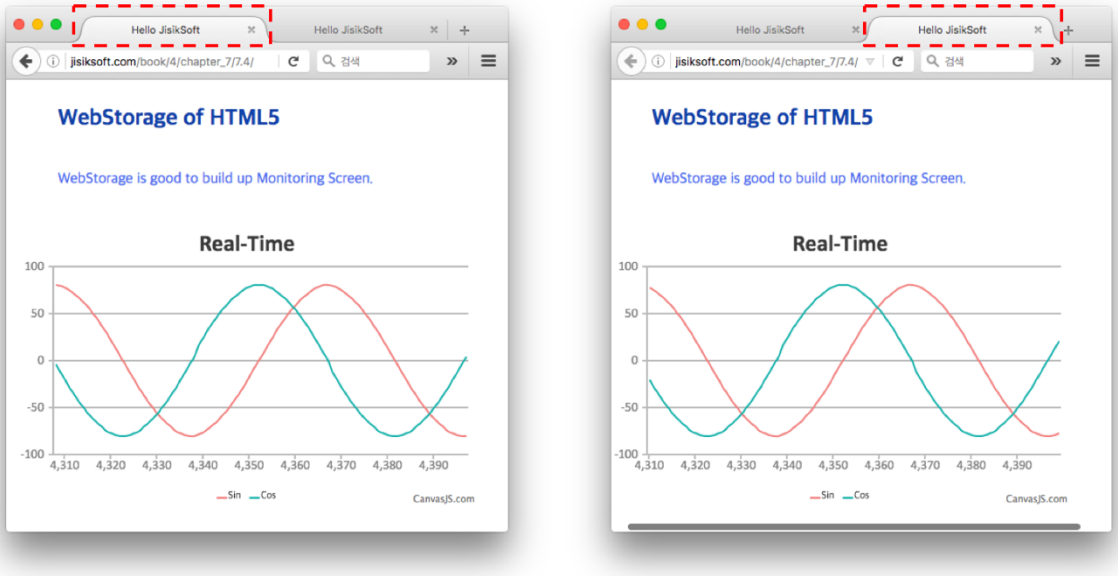
7.4 Web Storage

'Web Storage'는 JavaScript 언어의 변수를 선언해서 저장하는 대신 브라우저에 저장공간을 만들어서 데이터를 보관한다. 브라우저에서 '새로고침'(Refresh) 버튼을 클릭하면 모든 데이터를 서버에서 다시 받아서 초기화 작업을 진행하는데 이때 변수의 기존 데이터는 없어지고 초기화된다. 그러나, 'Web Storage'를 사용하면 데이터를 브라우저에서 보관하고 있기 때문에 기존의 데이터를 다시 사용할 필요가 있는 웹 프로그래밍에서는 상당히 유용한 HTML5의 기능이다. 'Web Storage' 기능을 여러 곳에서 사용할 수 있는데, 가장 많이 사용하는 곳이 '모니터링' 화면을 만들 때 사용한다. 시스템의 정보나 서비스를 제공하는 소프트웨어의 데이터는 실시간으로 변하는데, 브라우저에서 제공하는 모니터링 페이지는 실시간으로 변하는 서버의 데이터를 주기적으로 받아서 'Web Storage'에 저장하고 그래프로 보여준다. 모니터링 페이지는 해당 데이터를 가지고 다른 형태의 그래프를 보여주는 화면으로 이동할 수 있는데, 'Web Storage'에서 데이터를 보관하기 때문에 이전 데이터들을 가지고 그래프를 그릴수 있다. 'Web Storage'에는 'Session Storage'와 'Local Storage'로 나누어서 사용하는데, 'Session Storage'는 브라우저의 하나의 탭Tab 화면을 위해서 데이터가 보관되고 'Local Storage'는 탭에 상관없이 브라우저 전체에서 데이터를 보관한다. 즉, 'Session Storage'는 하나의 탭 화면에서 다른 페이지로 이동해도 데이터가 보존되며, '새로고침'(Refresh)을 클릭하면 데이터가 없어지는 변수와 달리 이전 데이터를 사용할 수 있지만, 탭 화면을 닫으면 'Session Storage'의 데이터는 사라진다. 반면에, 'Local Storage'에 저장된 데이터는 하나의 브라우저에 다수의 탭 화면이 존재해도 하나의 'Local Storage'에 저장된 데이터를 모든 탭들이 공유한다.

[그림 7-6]은 'Local Storage'를 사용해서 서버로부터 받은 데이터를 서로 다른 탭에서 동시에 보여주는 화면을 보여주는데, 새로운 탭을 만들어서 실행시켜도 모든 탭들이 기존의 데이터를 사용해서 동일한 그래프를 보여준다.

확인 사이트: http://jisiksoft.com/book/4/chapter_7/7.4/webstorage.html

그림 7-6 localStorage 를 사용해서 다수의 탭에서 같은 데이터를 공유할 수 있다.

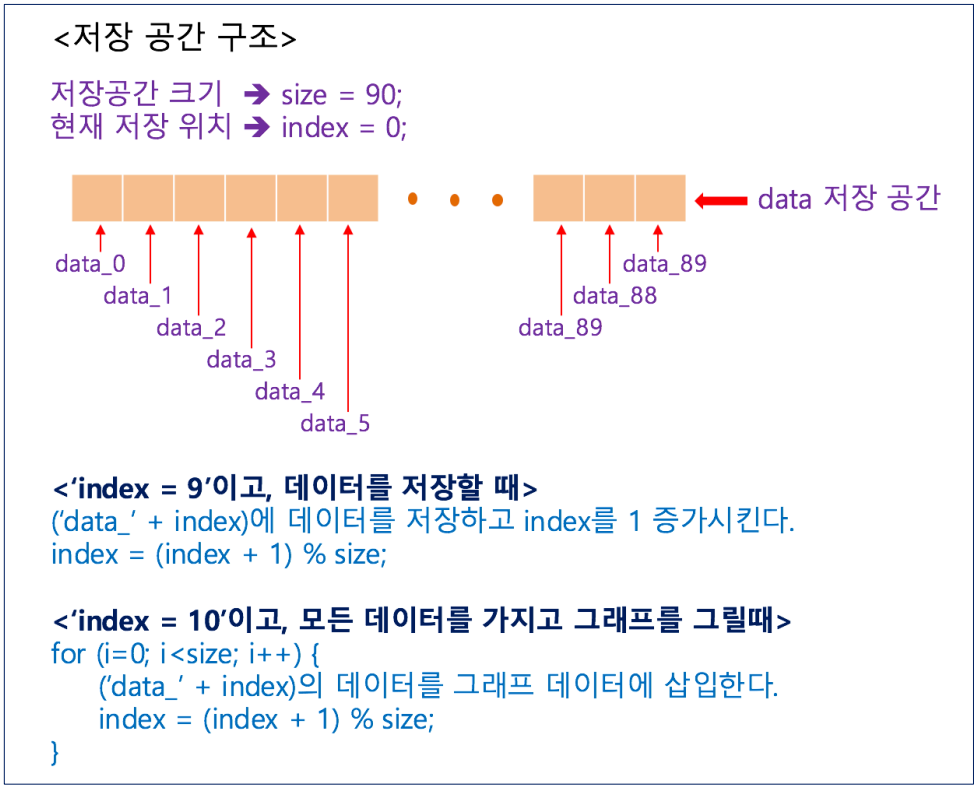


'Session Storage'와 'Local Storage'에 저장하는 데이터는 모두 문자열String 형식이기 때문에 JavaScript 언어에서 사용하는 데이터는 문자열로 변환해서 저장한다. 예를 들면, 배열 형식의 데이터를 'Local Storage'에 저장할 때는 배열 형식의 문자열로 변환하고, 데이터를 사용할 때는 문자열을 배열 형식으로 치환해서 사용한다. 그래서 'Local Storage'에서는 하나의 저장공간을 만들어서 다수의 데이터를 한번에 저장하는 방법보다는 다수의 저장공간을 만들어서 작은 데이터들을 저장해서 사용하는 것이 좋다.

[그림 7-7]은 이번 장에서 사용하는 'Local Storage'의 저장 공간 구조를 보여주는데, 90개 배열형태의 저장공간은 'data_0'부터 'data_89'까지의 이름을 가지고 있다. 90개의 저장공간은 배열이 아니지만, 공간에 접근하기 위한 이름을 일정한 형식으로 만들어서 배열과 같은 개념으로 만들었다. 또한, 'index'라는 저장공간에 숫자를 저장해서 현재 'data 저장 공간'에서 순서를 정하는데, 'index'에 저장되는 값은 '0'부터 '89'까지의 숫자들이다. 'index'의 값을 가지고 배열 형태의 'data 저장 공간'을 '원형 리스트'(Circular-List) 형식의 자료구조를 만들수 있는데, 'index' 값은 1씩 순차적으로 증가하고 마지막 값인 '89'까지 증가한 이후의 다음 증가값은 '0'이 된다. 이와 같은 형식은 "index = (index + 1) % size" 연산으로 구현하는데, "size = 90;"이기 때문에 index는 순차적으로 증가하면서 0부터 89까지의 값만을 갖게 된다. 배열을 사용해서 '원형 리스트'(Circular-List)를 구현하는 자료구조의 기법을 이해하면 이후에 필요한 곳에서 적절히 사용할 수 있다. '원형 리스트'에 데이터를 추가하는 방법은 'index'의 위치에 자료를 저장하고 'index'를 1 증가시키고, 데이터를 가져올 때는 'index'의

위치에 저장된 값이 가장 오래전 데이터이고 'index'를 증가하면서 데이터를 받은 순서대로 저장된 데이터를 읽어온다.

그림 7-7 배열 형식의 저장공간을 Circular 구조로 사용해서 90 개의 데이터들을 저장하는 구조



[코드 7-6]과 [코드 7-7]은 'Local Storage'를 사용해서 데이터를 저장하고 그래프를 그리는 프로그램을 만드는 코드이며, '3.3장'에서 사용한 'CanvasJS' 라이브러리를 사용해서 그래프를 그린다. 또한 'Local Storage'를 사용해서 구현하는 프로그램은 브라우저의 다수의 탭Tab 화면에서 동시에 같은 그래프를 볼 수 있음을 확인해야 한다. 'Local Storage'를 사용할 때 중요한 점은 다수의 '탭'에서 공통된 'Local Storage'를 사용하기 때문에 'Local Storage'에 데이터를 저장할 때는 하나의 '탭'에서만 저장하도록 JavaScript 언어로 구현해야 한다. 하나의 탭에서만 서버로부터 데이터를 가져와서 'Local Storage'에 저장하기 위해서는 'Local Storage'에 데이터를 가져온 시간을 저장해서 관리하면 되는데, 여기서는 가장 먼저 실행해서 데이터를 가져오는 탭의 프로그램은 2초마다 데이터를 주기적으로 가져와서 현재의 시간과 데이터를 'Local Storage'에 저장하고 이후에 생성되는 다른 탭의 코드에서는 2.5초마다 시간을 주기적으로 확인해서 저장된 시간과 현재 시간의 차이가 2.5초 미만이면 데이터를 가져오지 않는다. 이와 같은 방법으로 하나의 탭에서만 서버로부터 데이터를 가져오게 구현할 수 있으며, 데이터를

가져오는 코드의 탭을 브라우저에서 종료하면 다른 탭의 코드가 자동적으로 데이터를 가져오는 작업을 수행한다. [코드 7-7]에서 'Local Storage'를 사용하기 위해서 'localStorage'를 사용해서 코드를 만들었는데, 만약 하나의 탭에서만 자료를 저장하기 위해서 'Session Storage'를 사용한다면 모든 'localStorage'를 'sessionStorage'로 변경하면 된다. 'localStorage'와 'sessionStorage'의 단어변경은 대부분의 프로그래밍 편집기Editor에서 일괄적으로 모든 단어를 변경하는 기능을 제공하기 때문에 간단히 변경할 수 있다.

[코드 7-6] 서버로부터 데이터를 가져와서 챠트를 그리기 위한 HTML 코드 (webstorage.html)

```
<!DOCTYPE html>

<head>
	<title>Hello JisikSoft</title>
	<meta charset="utf-8"></meta>
	<link rel="stylesheet" href="./css/webstorage.css"></link>
	<script src="./js/jquery-2.1.3.min.js"></script>
	<script src="./js/canvasjs.min.js"></script>                    //---①
	<script src="./js/webstorage.js"></script>
</head>
<body>
	<br>
	<div id='subject'>WebStorage of HTML5</div>
	<br><br>
	<div id='contents'>WebStorage is good to build up Monitoring Screen.</div>
	<br><br>
	<div id="chart"></div>                                          //---②

	<script>
	   $(window).bind("mousedown", function() {
	            return false;
	   });

	   startMonitoring("chart");                                    //---③
	</script>
</body>
</html>
```

① 그래프 라이브러리로 'CanvasJs'를 사용하기 위해서 'canvasjs.min.js' 파일을 연결했다.

② 실시간 챠트를 그리기 위해서 'chart'라는 id 값을 가진 태그를 추가했으며, 해당 태그의 크기는 'webstorage.css' 파일에서 디자인 속성으로 정의되었

다.

③ 'Local Storage'를 사용해서 서버의 실시간 데이터를 저장하고 실시간 챠트를 그리는 함수를 실행한다.

[코드 7-7] 서버에서 데이터를 받아서 localStorage 에 저장하고 챠트를 그린다. (/js/webstorage.js)

```
var chart = null;                                                    //---①
var size = 90;
var flagToGetData = false;
var timeToDrawChart = 2;

function startMonitoring( chartContainer ) {                         //---②

    if (typeof(Storage) !== null) {                                  //---③

        if ( localStorage.getItem("time") == null ) {                //---④

            var currTime = new Date().getTime();                     //---⑤

            localStorage.setItem("count", size);                     //---⑥
            localStorage.setItem("index", 0);
            localStorage.setItem("time", currTime);

            for (var i=0; i<size; i++) {                             //---⑦
                localStorage.setItem("data_"+i, '{"sin":0, "cos":0}');
            }

            flagToGetData = true;                                    //---⑧
        }

        createChart(chartContainer);                                 //---⑨

        getRealTimeData();                                           //---⑩
        drawRealTimeChart();                                         //---⑪

    } else {
        alert("브라우저가 HTML5를 지원하지 않습니다.");
    }

}
```

```
function createChart(chartContainer) {                                    //---⑫

        chart = new CanvasJS.Chart( chartContainer, {
            title:{     text: "Real-Time",
                        fontSize: 25
            },
            data: [{  type: "spline",
                        showInLegend: true,
                        lineThickness: 2,
                        name: "Sin",
                        markerType: "square",
                        color: "#F08080",
                        dataPoints: getChartData(0)
            },
            {           type: "spline",
                        showInLegend: true,
                        lineThickness: 2,
                        name: "Cos",
                        color: "#20B2AA",
                        dataPoints: getChartData(1)
            }]
        });

      chart.render();
}

function getChartData(no) {                                               //---⑬

        var count = localStorage.count * 1;
        var index = localStorage.index;

        var result = new Array();
        var i, jsonData, value;

        for(i=0; i<size; i++) {

            if (typeof localStorage["data_"+index] !== null
                        && localStorage["data_"+index] !== null) {
                        jsonData = JSON.parse(localStorage["data_"+index]);
            }
            if (no == 0) {
                        value = jsonData.sin;
            } else if (no == 1) {
                        value = jsonData.cos;
            }
            result.push( {x: count-(size-i+1), y:value} );

            index = ((index * 1) + 1) % size;
        }

        return result;
```

```
}

function drawRealTimeChart() {                                          //---⑭

        chart.options.data[0].dataPoints = getChartData(0);
        chart.options.data[1].dataPoints = getChartData(1);

        chart.render();

        setTimeout( drawRealTimeChart, (timeToDrawChart * 1000) );
}

function addStorageData( data ) {                                       //---⑮

        localStorage.count = (localStorage.count * 1) + 1;
        localStorage.time = new Date().getTime();
        localStorage["data_"+localStorage.index] = data;
        localStorage.index = ((localStorage.index * 1) + 1) % size;
}

function getRealTimeData() {                                            //---⑯

        if (flagToGetData) {
            var data = doAjax('./php/ajax_get_data.php');
            addStorageData( data );

            setTimeout( getRealTimeData, 2000 );
            return;
        }

        var currTime = new Date().getTime();

        if ((currTime - localStorage.time) > 2500) {
            flagToGetData = true;
        }
        setTimeout( getRealTimeData, 2500 );
}

function doAjax(str_url, input_data) {
        var result;
        $.ajax({
                    url: str_url,
                    type: 'post',
                    async: false,
                    datatype: 'json',
                    data: input_data,
                    error: function() {
                                alert('Network Error : 서버와의 연결에 문제가 있습니다.');
                    },
```

```
                success: function(obj) {
                        result = obj;
                }
        });
        return result;
}
```

① 챠트를 그리기 위해서 'CanvasJS' 라이브러리를 사용해서 생성되는 챠트 객체를 위해서 'chart' 변수를 만들었으며, 2초마다 그래프를 다시 그리기 위해서 'chart' 변수로 객체에 접근한다. 'size' 변수는 챠트에 그리는 데이터의 갯수를 지정하기 위해서 사용한 변수이며 90으로 설정되어서 90개의 데이터를 챠트에서 그래프로 보여주고 'Local Storage'에 저장되는 데이터도 90개이다. 'flagToGetData' 변수는 데이터를 서버에서 가져오면 true 값을 가지고, 만약 'false'이면 서버에 데이터를 요청하지 않는다. 'timeToDrawChart' 변수는 그래프를 그리는 간격의 초단위의 값을 갖는데, 'timeToDrawChart' 변수의 값이 2이기 때문에 2초마다 그래프를 새로운 데이터로 다시 그린다.

② startMonitoring() 함수는 실시간으로 서버에서 데이터를 가져와서 'Local Storage'에 저장하고, 저장된 데이터의 챠트를 그린다. 함수의 매개변수로 전달받는 'chartContainer' 변수는 챠트를 그리는 태그의 id 값을 갖는다.

③ HTML5가 발표되기 전에 만들어진 브라우저들은 'Web Storage' 기능을 제공하지 않기 때문에 'Storage'가 정의된 브라우저에서만 'localStorage'를 사용하고 그렇지 않으면 경고 창으로 HTML5를 지원하지 않는 브라우저라고 사용자에게 알려준다.

④ 'localStorage'에 현재시간을 저장하는 'time'을 위한 공간이 만들어지지 않았다면, 현재 프로그램을 브라우저에서 처음 실행하기 때문이며 'localStorage'에 필요한 공간들을 만드는 작업을 수행한다. 이후에 실행되는 프로그램은 'localStorage'에 'time' 값이 존재하기 때문에, 'localStorage'에 저장된 데이터를 가지고 챠트를 그린다.

⑤ 현재시간의 milliseconds(1/1000-초 단위) 값을 가져오기 위해서 Data() 클래스의 getTime() 함수를 실행해서 결과값을 currTime 변수에 저장한다.

⑥ 'localStorage'에 필요한 공간을 만들기 위해서 setItem() 함수를 사용한다. 그래프의 x-축에 숫자를 넣기 위해서 'count' 항목을 사용하며, 초기에 90개의 데이터를 그래프에 넣기 위해서 size(= 90)를 초기화한다. [그림 7-7]에서 설명한 방법과 같이 서버에서 받아오는 데이터는 배열 형태의 데이터 공간에 저장하는데, 저장되는 데이터의 순서를 정하기 위해서 'index' 항목을 사용하고 초기값으로 '0'을 갖는다. 현재 시간을 저장하기 위해서 'time' 항목을 'localStorage'에 만들고, 현재 시간 값을 저장한다.

⑦ 90개의 데이터를 저장하기 위해서 'data_0'부터 'data_89'까지의 항목을 'localStorage'에 만들고, 모든 항목의 초기값은 '{ "sin":0, "cos":0 }'이라는 JSON 데이터 형식의 문자열을 저장한다. 서버로부터 받아오는 형식은 삼각함수의 Sin과 Cos 값을 가지는 JSON 형식의 문자열이기 때문에, 모든 데이터 공간을 0의 값을 가지는 문자열로 초기화했다.

⑧ 'localStorage'에 데이터 공간을 만드는 과정들이 진행된 이후에 서버로부터 데이터를 가져오기 위해서 'flagToGetData' 변수를 true로 설정한다.

⑨ 'CanvasJS' 라이브러리의 클래스 생성자를 사용해서 챠트를 생성한다.

⑩ 서버로부터 실시간 데이터를 가져오기 위해서 getRealTimeData() 함수를 실행한다.

⑪ 실시간 챠트를 그리기 위해서 drawRealTimeChart() 함수를 실행한다.

⑫ 챠트를 생성하기 위해서 'CanvasJS'의 Chart 클래스를 생성하며, 부드러운 곡선의 챠트를 만들기 위해서 type을 'spline'으로 설정했다. 챠트에 삽입되는 데이터는 두 개이며, 'data' 항목에 배열로 이루어진 두 개의 JSON 형식의 데이터 설정값들을 만들었다. 'CanvasJS' 라이브러리에 대한 자세한 설명은 'CanvasJS 홈페이지'(http://canvasjs.com)에서 확인할 수 있다.

⑬ getChartData() 함수는 'Local Storage'에 저장된 데이터를 가져와서 챠트에 삽입되는 데이터를 만드는데, 매개변수인 'no'의 값이 0이면 챠트의 첫 번째 데이터인 Sin 데이터를 JSON 형식의 값을 갖는 배열로 만들고, 1이면 두 번째 데이터인 Cos 데이터를 배열로 만든다. 함수에서 count 변수는 챠트에서 x-축에 나타나는 정수값을 만들기 위해서 사용하고, index 변수는 배열 형식으로 저장된 데이터들을 '원형 리스트' 형식으로 읽어오기 위해서 사용한다. [그림 7-7]의 아래부분에서 설명한 방법과 같이 'index'를 사용해서 데이터를 차례대로 읽어오는데, for loop의 아래에 "index = ((index * 1) + 1) % size;"식은 index를 1씩 증가하는 연산이고 "(index * 1)"의 연산을 수행하는 이유는 문자열을 숫자로 변경하기 위해서 1을 곱한 것이다.

⑭ 실시간 챠트를 그리기 위해서 실행하는 drawRealTimeChart() 함수는 getChartData() 함수를 사용해서 챠트에 삽입하는 데이터를 'Local Storage'로부터 만들어서 챠트에 넣어주고 render() 함수를 사용해서 챠트를 다시 그려준다. timeToDrawChart 변수에는 2가 저장되었기 때문에 setTimeout() 함수를 사용해서 2초마다 drawRealTimeChart()를 다시 실행해서 주기적으로 실시간 챠트를 그린다.

⑮ addStorageData() 함수는 매개변수로 받는 data를 'Local Storage'에 저장한다. 현

재 저장하는 데이터가 몇 번째인지를 카운트하기 위해서 'localStorage'의 'count' 항목을 1 증가시키고, 'time' 항목도 현재 시간으로 변경한다. 또한 'index' 항목의 값에 맞는 곳에 데이터를 삽입하고, 'index' 항목을 1 증가시킨다.

⑯ 만약 flagToGetData 변수가 true이면 2초마다 서버의 './php/ajax_get_data.php' 파일을 실행해서 실시간 데이터를 받고 addStorageData() 함수를 실행해서 'localStorage'에 데이터를 저장한다. 하지만, flagToGetData 변수가 true가 아니면, 서버로부터 데이터를 가져오지 않고 2.5초 뒤에 getRealTimeData() 함수를 다시 실행한다. 만약 현재시간과 'localStorage'의 'time' 항목의 시간이 2.5초 이상의 차이가 생기면 서버로부터 데이터를 받아오는 이전에 실행된 프로그램이 작동을 안하는 것이기 때문에 flagToGetData 변수를 true로 설정해서 이후부터는 서버로 데이터를 요청하게 된다.

[코드 7-8]은 브라우저로 데이터를 전송하는 서버의 프로그램을 간단히 만들기 위해서 삼각함수의 Sin과 Cos 값들을 계산해서 JSON 형식의 데이터로 만드는 PHP 코드이다. 삼각함수의 값들은 현재시간을 기준으로 만드는데, 현재시간의 초단위 값을 반환하는 time() 함수를 사용해서 삼각함수를 만들기 위한 각도Degree를 계산하고, 반지름이 80인 원의 Sin과 Cos 값들을 만들어서 브라우저로 전송한다.

[코드 7-8] Sin 과 Cos 값을 Jason 형식의 문자열로 만든다. (/php/ajax_get_data.php)

```
<?php

$radius = 80;                                                    //---①
$degree = (((time() * 3) % 360) * M_PI) / 180;                   //---②

$sin = round(sin($degree) * $radius);                            //---③
$cos = round(cos($degree) * $radius);

print '{ "sin":'.$sin.', "cos":'.$cos.' }';                      //---④
exit();

?>
```

① 삼각함수는 원을 기준으로 계산하는 함수이기 때문에 반지름이 80인 원의 Sin과 Cos 값을 만들기 위해서 $radius 값을 80으로 설정했다.

② $degree 변수는 삼각함수를 계산하는 각도를 저장하는데, 현재시간을 초값으로

전달하는 time() 함수와 원의 pi(파이) 값을 갖는 상수변수 M_PI를 사용해서 현재시간의 각도를 계산하다.

③ 현재시간을 기준으로 계산된 각도($degree)를 기준으로 sin()과 cos() 함수의 값을 계산해서 $sin과 $cos 변수에 저장한다.

④ 삼각함수의 Sin과 Cos 값들을 JSON 형식의 문자열로 만들어서 브라우저로 전송한다.

프로그래밍을 잘하기 위해서 어떤 과목을 공부해야 하는지에 대한 질문을 받은 적이 있다. 어떠한 학문을 공부해도 프로그래밍을 자신의 분야와 접목시키면 참신한 프로그램을 만들 수 있는데, 프로그래밍을 하다보면 고등학교 때까지 공부한 수학적 배경지식을 필요로 할 때가 많다는 것을 느낀다. 이번 장에서 삼각함수 그래프를 만들기 위해서 각도를 계산하는 방법도 인터넷에서 참고할 수 있지만, 참고한 자료를 자신만의 수식으로 변경하기 위해서는 삼각함수에 대한 배경지식을 필요로 한다. 예를 들면, 7.2장에서 다룬 3D 프로그래밍을 할 때는 기하학적으로 공간의 구조를 잘 파악하는 사람들이 (x, y, z) 좌표를 가지고 계산하는 수식을 쉽게 만들수 있다. 물론, 필자도 아직은 3D 프로그래밍을 어렵게 느끼는데, 3D 프로그래밍만을 전문으로 하는 사람들은 많은 시간을 가지고 충분한 연습을 했을 것이다. 이 책에서와 같이 'three.js' 라이브러리를 사용해서 간단한 프로그램을 만들 수도 있지만, 'three.js' 같은 라이브러리를 만드는 사람들은 3D 프로그래밍을 하기 위한 수학적 배경지식이 상당히 높을 것이다.

7.5 Web Worker

모든 운영체제는 멀티태스킹(MultiTasking) 기능을 제공하는데, 컴퓨터로 음악을 들으면서 동시에 여러 개의 소프트웨어들을 사용해서 작업을 진행할 수 있다. 컴퓨터의 사양을 보면 "Dual Core CPU"라고 나오는데, 이는 컴퓨터 안에 꽂힌 한 개의 CPU에는 독립적으로 구분된 2개의 CPU들이 동시에 동작한다는 개념이다. 4를 의미하는 Quad 단어가 붙어서 "Quad Core CPU"라고 하면 한 개의 CPU에는 독립적으로 구분된 4개의 CPU가 존재하고 동시에 동작할 수 있다는 개념이다. 아주 복잡한 연산을 수행하는 프로그램을 오랫동안 실행해도 "Quad Core CPU"에서는 4개 중에

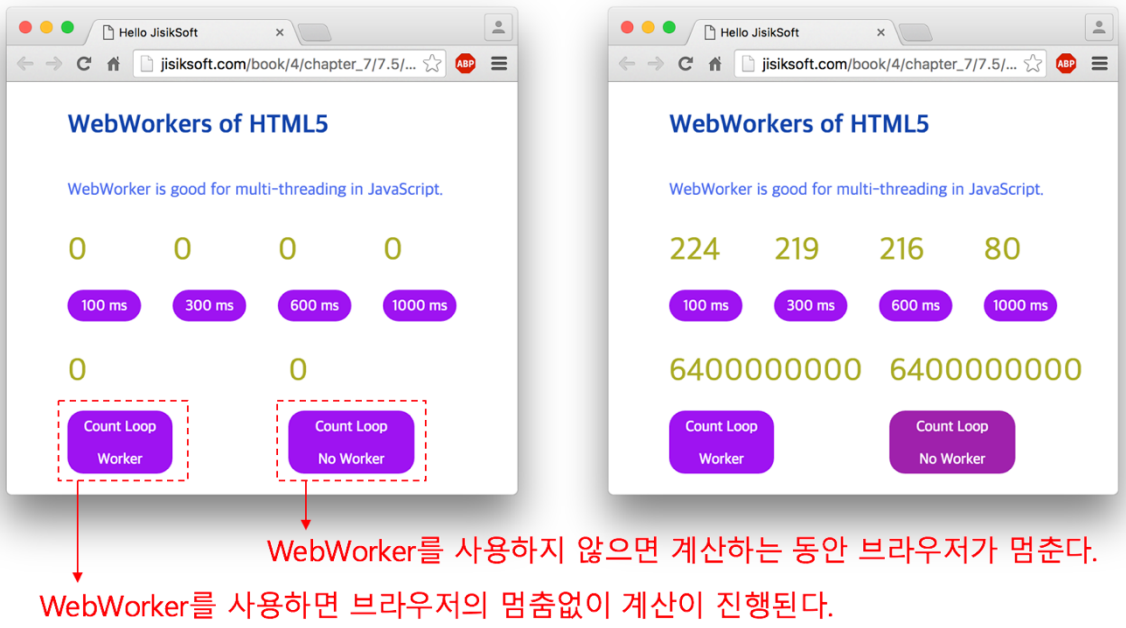
1개만 사용하기 때문에 CPU 사용률이 25%이상 올라가지 않는다. 운영체제는 자신이 사용하는 CPU의 갯수를 알기 때문에 동시에 여러 개의 소프트웨어를 실행해도 다수의 CPU에 개별적으로 일을 부여한다. 만약 "Dual Core CPU"에서 동시에 4개의 소프트웨어를 실행한다면 운영체제는 2개의 CPU에서 각각 2가지 작업들을 동시에 진행할 수 있게 한다. 멀티태스킹(Multitasking)이 운영체제에서 동시 작업을 수행하는 개념이라면 소프트웨어적으로 동시에 다수의 일을 수행할 수 있게 만든 것이 멀티쓰레딩(MultiThreading) 기능이다. C/C++이나 Java와 같은 대부분의 프로그래밍 언어에서는 멀티쓰레딩을 사용해서 프로그램을 만들 수 있는데, 하나의 프로그램에서 다수의 작업을 동시에 진행할 수 있다. 예를 들면, TCP/IP 통신을 하는 서버측 프로그램을 만든다면, 하나의 쓰레드Thread는 항상 클라이언트Client의 접속을 기다리는 역할을 하고 이미 연결이 진행된 항목들은 다수의 쓰레드Thread들로 만들어져 개별적으로 사용자와 통신을 진행한다. 즉, 운영체제에서는 하나의 태스크Task를 생성해서 하나의 소프트웨어를 실행하지만, 하나의 소프트웨어는 다수의 쓰레드들Threads을 만들어서 동시에 여러 작업들을 수행한다. 하나의 프로그램이 하나의 CPU에서 동작한다면 프로그램은 다수의 쓰레드들을 만들어서 각각의 쓰레드의 작업을 차례대로 조금씩 수행하는데 CPU가 아주 빠르게 처리하기 때문에 사용자는 마치 동시에 다수의 쓰레드들이 움직이고 있다고 느끼게 된다. HTML5로 진화하면서 웹 프로그래밍에서도 멀티쓰레드를 사용하기 위해서 만든 것이 'Web Worker' 기능이다. 웹 프로그램은 브라우저에서 HTML, CSS, JavaScript 코드들을 해석하고 실행하는데, 'Web Worker' 기능을 수행하는 브라우저는 멀티쓰레드를 만들어서 동시에 여러 작업들을 수행하는 효과를 구현했다. 하나의 탭에서 보는 화면을 하나의 쓰레드라고 하면 화면에서 보이지 않는 곳에서 다수의 쓰레드들을 만들어서 동시에 다른 기능들을 수행한다.

[그림 7-8]은 'Web Worker'를 사용해서 다양한 연산을 수행하는 다수의 쓰레드들을 만들었는데, 멀티쓰레드의 동작을 이해하기 위해서는 화면의 윗쪽에 있는 4개의 버튼을 차례대로 클릭한 후에 아래쪽 두 개의 버튼들을 클릭하면 쉽게 확인할 수 있다. 윗쪽의 4개의 버튼과 아래의 왼쪽에 있는 버튼들은 모두 'Web Worker'의 쓰레드에서 동작하기 때문에 화면의 멈춤 없이 연산들을 수행하지만, 아래의 오른쪽에 있는 버튼을 클릭하면 화면의 쓰레드에서 연산을 수행하기 때문에 연산을 하는 동안에 화면이 5초이상 멈춘다. HTML5를 지원하는 브라우저는 하나의 CPU를 사용하

지만, 5개의 쓰레드들을 추가로 생성해서 화면의 쓰레드를 포함한 6개의 쓰레드들을 짧은 시간 동안에 차례대로 돌아가면서 동작을 수행하기 때문에 사용자는 동시에 6개의 쓰레드들이 움직이고 있다고 느끼게 된다. 웹 프로그래밍을 하면서 멀티쓰레딩 기능을 사용하는 경우는 많지 않지만, 필요할 때 멀티쓰레딩을 사용할 수 있다는 것은 사용자가 불편함을 느끼지 않고 게임과 같은 무거운 웹 프로그램을 개발할 수 있는 환경이 만들어졌다고 이해할 수 있다. 이 책에서 'Web Socket'을 다루지는 않지만, HTML5의 'Web Worker'와 함께 'Web Socket'을 활용해서 기존의 온라인 게임을 브라우저에서 실행할 수 있게 만드는 것도 가능하다고 생각한다.

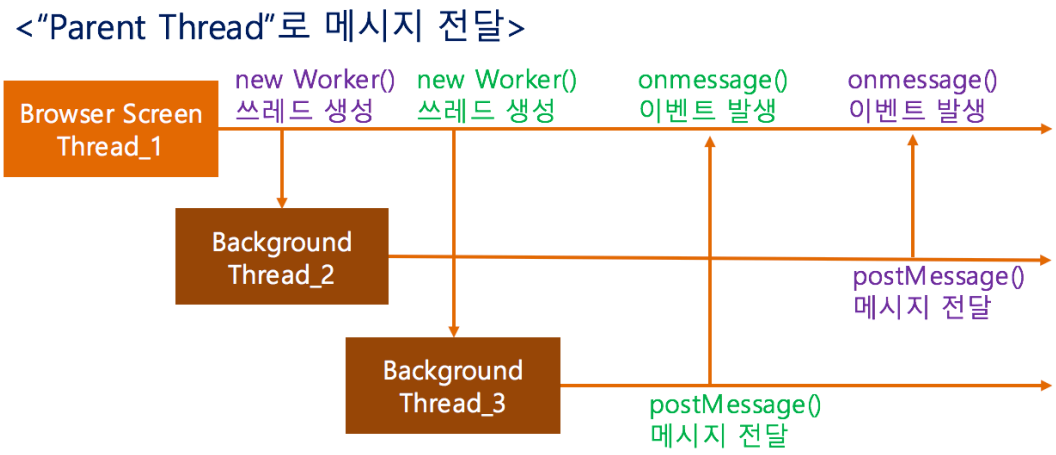
확인 사이트: http://jisiksoft.com/book/4/chapter_7/7.5/webworkers.html

그림 7-8 '멀티쓰레드'로 동작하는 WebWorker 를 사용해서 브라우저의 멈춤 없이 연산이 가능하다.



[그림 7-9]는 'Web Worker'에서 생성된 쓰레드에서 메시지를 전달하는 그림을 보여주는데, 브라우저 화면의 쓰레드에서 두 개의 쓰레드들을 생성했으며 화면 쓰레드는 부모 쓰레드Parent Thread이고 두 개의 백그라운드Background 쓰레드들은 자식 쓰레드Child Thread가 된다. 쓰레드 간의 메시지 전달은 postMessage()를 사용해서 보내고, 부모 쓰레드에서 onmessage() 이벤트가 발생해서 메시지를 전달 받는다.

그림 7-9 '멀티쓰레드'의 메시지는 쓰레드를 생성한 부모로 전달된다.



브라우저로 웹 프로그램을 실행하는 사용자는 브라우저의 화면만 보기 때문에 백그라운드 쓰레드에서 발생하는 데이터들을 확인하기 위해서는 onmessage() 이벤트를 통해서 전달받은 내용을 브라우저의 화면에 출력해서 확인할 수 있다. [코드 7-9]는 'Web Worker'를 사용해서 백그라운드에서 연산을 수행하고 결과를 전달받아 화면에 주기적으로 보여주는 프로그램의 HTML 코드이며, 연산의 결과를 화면에 보여주기 위한 태그들과 버튼을 만들기 위한 태그들을 만들었다.

[코드 7-9] '멀티쓰레드'를 실행하는 화면을 만드는 HTML 코드 (webworkers.html)

```
<!DOCTYPE html>

<head>
	<title>Hello JisikSoft</title>
	<meta charset="utf-8"></meta>
	<link rel="stylesheet" href="./css/webworkers.css"></link>
	<script src="./js/jquery-2.1.3.min.js"></script>
	<script src="./js/webworkers.js"></script>
</head>
<body>
	<br>
	<div id='subject'>WebWorkers of HTML5</div>
	<br><br>
	<div id='contents'>WebWorker is good for multi-threading in JavaScript.</div>
	<div class="lineEmpty"></div>
	<div class="text" id="text1">0</div>                                    //---①
	<div class="text" id="text2">0</div>
	<div class="text" id="text3">0</div>
	<div class="text" id="text4">0</div>
```

```
        <div class="lineEmpty"></div>
        <div class="button" id="btn1" onclick="startWorker(1, 'text1');">100 ms</div>   //---②
        <div class="button" id="btn2" onclick="startWorker(2, 'text2');">300 ms</div>
        <div class="button" id="btn3" onclick="startWorker(3, 'text3');">600 ms</div>
        <div class="button" id="btn4" onclick="startWorker(4, 'text4');">1000 ms</div>
        <div class="lineEmpty"></div>
        <div class="text" id="text5">0</div>
        <div class="text" id="text6">0</div>
        <div class="lineEmpty"></div>
      <div class="button" id="btn5" onclick="startWorker(5, 'text5');">
                    Count Loop<br>Worker</div>                                    //---③
      <div class="button" id="btn6" onclick="countLoop('text6');">
                    Count Loop<br>No Worker</div>
</body>
</html>
```

① 생성되는 다수의 쓰레드로부터 받은 결과값을 화면에 출력하기 위해서 'text'라는 클래스 값을 가진 태그들을 만들었다.

② 백그라운드에서 진행되는 연산을 하기 위해서 startWorker() 함수를 실행하는 버튼들을 만들기 위해서 'button'이라는 클래스 값을 가진 태그들을 만들었다.

③ 'Count Loop Worker' 버튼과 'Count Loop No Worker' 버튼은 동일한 연산을 수행하지만, 첫 번째 버튼은 백그라운드 쓰레드에서 연산을 하고 두 번째 버튼은 화면 쓰레드에서 연산을 수행한다.

[코드 7-10]은 화면의 버튼이 클릭되었을 때 실행하는 JavaScript 함수들이며, 'Web Worker'를 실행하는 startWorker() 함수는 화면 쓰레드 함수이고 Worker() 생성자로 만드는 새로운 클래스들은 모두 백그라운드 쓰레드들이다. 생성된 클래스의 onmessage() 이벤트 함수는 백그라운드 쓰레드들이 전달하는 메시지를 받을 때마다 실행되며, 함수의 event의 데이터를 화면에 출력한다.

[코드 7-10] '멀티쓰레드'를 실행하는 Javascript 코드 (/js/webworkers.js)

```
var w1, w2, w3, w4, w5;                                                    //---①

function startWorker(no, container) {                                      //---②
      if(typeof(Worker) !== "undefined") {                                 //---③
         if (no == 1) {                                                    //---④
               if(typeof(w1) == "undefined") {                             //---⑤
```

```
                w1 = new Worker("./js/worker1.js");                    //---⑥
                w1.onmessage = function(event) {                       //---⑦
                        $('#'+container).html(event.data);
                }
            }
        } else if (no == 2) {
            if(typeof(w2) == "undefined") {
                w2 = new Worker("./js/worker2.js");
                w2.onmessage = function(event) {
                        $('#'+container).html(event.data);
                }
            }
        } else if (no == 3) {
            if(typeof(w3) == "undefined") {
                w3 = new Worker("./js/worker3.js");
                w3.onmessage = function(event) {
                        $('#'+container).html(event.data);
                }
            }
        } else if (no == 4) {
            if(typeof(w4) == "undefined") {
                w4 = new Worker("./js/worker4.js");
                w4.onmessage = function(event) {
                        $('#'+container).html(event.data);
                }
            }
        } else if (no == 5) {
            if(typeof(w5) == "undefined") {
                w5 = new Worker("./js/worker5.js");
                w5.onmessage = function(event) {
                        $('#'+container).html(event.data);
                }
            }
        }
    } else {
        alert("브라우저가 HTML5를 지원하지 않습니다.");
    }
}

function countLoop(container) {                                        //---⑧

    var count = 0;

    for (var i=0; i<80000; i++) {
        for (var j=0; j<80000; j++) {
                count += 1;
        }
    }
    $('#'+container).html(count);
```

```
}
```

① 쓰레드를 생성하기 위해서 Worker() 생성자를 사용하는데, Worker() 생성자로 생성된 5개의 Worker 클래스들을 가리키기 위해서 5개의 전역변수들을 만들었다.

② startWorker() 함수는 Worker() 클래스를 사용해서 쓰레드를 만드는데, 전달되는 첫 번째 매개변수 'no'는 몇 번째 쓰레드를 생성할지를 정하는 숫자이고 'container'는 생성된 쓰레드로부터 받은 결과값을 화면에 출력하기 위한 태그의 id 값이다.

③ 브라우저에서 'Web Worker' 기능을 지원하는지를 확인하고, 만약 지원하지 않는다면 HTML5를 지원하지 않는 브라우저라는 경고 창을 사용자에게 보여준다.

④ 'no'는 1부터 4까지의 정수값을 가지고 있는데, 'no'의 값에 따라서 하나의 백그라운드 쓰레드를 생성한다.

⑤ 첫 번째 Worker() 생성자를 실행하면 'w1' 변수는 생성된 클래스를 가리키고 있는데, 이후에 반복해서 같은 동작을 하는 쓰레드의 생성을 막기 위해서 'w1'이 생성되었는지를 확인한다.

⑥ Worker() 생성자를 실행해서 쓰레드를 만드는데, 생성자의 매개변수에는 파일 이름을 넣는다. 여기서는 './js/worker1.js' 파일의 JavaScript 코드를 백그라운드 쓰레드에서 실행한다. 이와 같은 방법으로 쓰레드에서 처리하는 프로그램 코드를 JavaScript 파일에 만들어서 관리한다.

⑦ 생성된 w1 클래스의 쓰레드에서 메시지를 전달하면 onmessage() 이벤트 함수가 자동으로 실행되고, 함수의 매개변수로 받는 event 변수에서 event.data에 저장된 메시지 내용을 화면에 출력한다.

⑧ 브라우저의 화면 쓰레드에서 실행되는 countLoop() 함수는 많은 연산을 수행하는데, 이 함수가 실행되는 동안에는 브라우저의 화면 쓰레드는 다른 이벤트를 받지 못한다. 그래서, 사용자는 브라우저의 동작이 잠시 멈춘 것으로 느끼게 된다.

[코드 7-11]부터 [코드 7-15]까지는 5개의 백그라운드 쓰레드에서 실행되는 JavaScript 코드들이며, 간단한 기능을 하는 함수를 만들어서 자동으로 실행한다. 백그라운드 쓰레드에서 수행하는 연산의 결과는 postMessage() 함수를 사용해서 자신을 생성한 부모 쓰레드로 전달된다.

[코드 7-11] 0.1 초마다 1 을 더하고 부모 쓰레드로 전달한다. (/js/worker1.js)

```
var count = 0;
```

```
function addCount() {                                          //---①
        count = count + 1;
        postMessage( count );
        setTimeout( addCount, 100 );
}

addCount();
```

① 0부터 시작하는 count 변수에 1을 더하고 postMessage() 함수를 사용해서 부모 쓰레드로 결과를 전달한다. setTimeout() 함수를 사용해서 0.1초마다 함수를 반복 실행한다.

[코드 7-12] 0.3 초마다 3 을 더하고 부모 쓰레드로 전달한다. (/js/worker2.js)

```
var count = 0;

function addCount() {                                          //---①
        count = count + 3;
        postMessage( count );
        setTimeout( addCount, 300 );
}

addCount();
```

① 0부터 시작하는 count 변수에 3을 더하고 postMessage() 함수를 사용해서 부모 쓰레드로 결과를 전달한다. setTimeout() 함수를 사용해서 0.3초마다 함수를 반복 실행한다.

[코드 7-13] 0.6 초마다 6 을 더하고 부모 쓰레드로 전달한다. (/js/worker3.js)

```
var count = 0;

function addCount() {                                          //---①
        count = count + 6;
        postMessage( count );
        setTimeout( addCount, 600 );
}

addCount();
```

① 0부터 시작하는 count 변수에 6을 더하고 postMessage() 함수를 사용해서 부모 쓰레드로 결과를 전달한다. setTimeout() 함수를 사용해서 0.6초마다 함수를 반복 실행한다.

[코드 7-14] 1 초마다 10 을 더하고 부모 쓰레드로 전달한다. (/js/worker4.js)

```
var count = 0;

function addCount() {                                          //---①
        count = count + 10;
        postMessage( count );
        setTimeout( addCount, 1000 );
}

addCount();
```

① 0부터 시작하는 count 변수에 10을 더하고 postMessage() 함수를 사용해서 부모 쓰레드로 결과를 전달한다. setTimeout() 함수를 사용해서 1초마다 함수를 반복 실행한다.

[코드 7-15] 백그라운드 쓰레드에서 연산을 수행하고 부모 쓰레드로 결과를 전송한다. (/js/worker5.js)

```
function countLoop() {                                         //---①

        var count = 0;

        for (var i=0; i<80000; i++) {
            for (var j=0; j<80000; j++) {
                        count += 1;
            }
        }

        postMessage( count );
}

countLoop();
```

① [코드 7-10]의 마지막 부분에 구현된 countLoop() 함수와 같은 연산을 수행하며, 'worker5.js' 파일에서 구현했기 때문에 백그라운드 쓰레드에서 실행되고 postMessage() 함수를 사용해서 부모 쓰레드로 결과를 전달한다.

필자는 다수의 웹 프로그래밍을 만들면서 'Web Worker'를 사용한 적은 없는데, 대부분의 웹 프로그램들이 사용자의 이벤트를 받아서 처리하는 것이 주요 목적이기 때문이었다. 새로운 쓰레드를 추가해서 사용자가 확인할 수 없는 동작을 수행하는 기능은 잘 사용하면 효과적이지만, 누군가 나쁜 용도로 사용한다면 위험할 수도 있는 기능이다. 어쨌든, 브라우저에서 멀티쓰레딩으로 웹 프로그래밍을 할 수 있게 만든 HTML5의 'Web Worker' 기능은 프로그래머에게는 아주 반가운 기능이다.

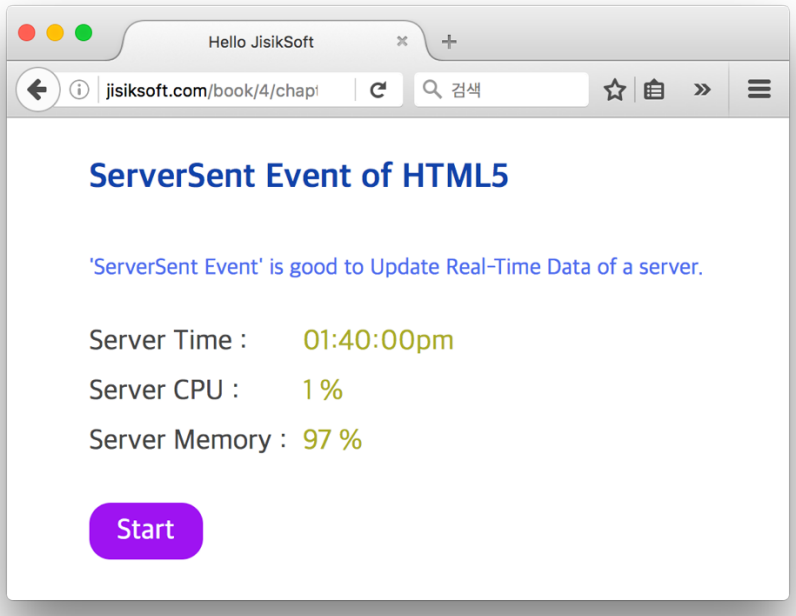
7.6 Server-Sent Events

HTML5의 'Server-Sent Events'는 서버에서 몇 초마다 주기적으로 라우저로 데이터를 전송하는 기능이다. 'Server-Sent Events'는 Facebook, Twitter, 주식, 뉴스, 스포츠 등에서 정보를 실시간으로 업데이트할 때 유용하게 사용된다고 하는데, 브라우저에서 데이터를 요청하는 것이 아니라 서버에서 데이터에 변화가 있을 때 데이터를 전송하므로 브라우저를 사용하는 사용자 입장에서 상당히 편리한 기능이다. 이번 장에서는 'Server-Sent Events' 기능을 사용해서 서버의 정보를 브라우저에서 모니터링하는 방법을 구현하는데, 웹 서비스를 하는 시스템에서 실시간으로 변경되는 데이터들을 브라우저에서 다수의 사용자가 모니터링할 때 유용하다. 'Server-Sent Events' 기능을 사용하면 브라우저는 서버와 연결을 유지한 상태에서 데이터를 계속 받게 되는데, 동영상을 받을 때 사용하는 스트리밍Streaming 서비스와 비슷한 개념으로 작은 데이터를 주기적으로 받는다.

[그림 7-10]은 'Server-Sent Events'를 사용해서 실시간 데이터를 받아서 화면에 출력하는 그림이며, 서버의 '현재 시간', 'CPU 사용률', 'Memory 사용률'을 실시간으로 보여준다.

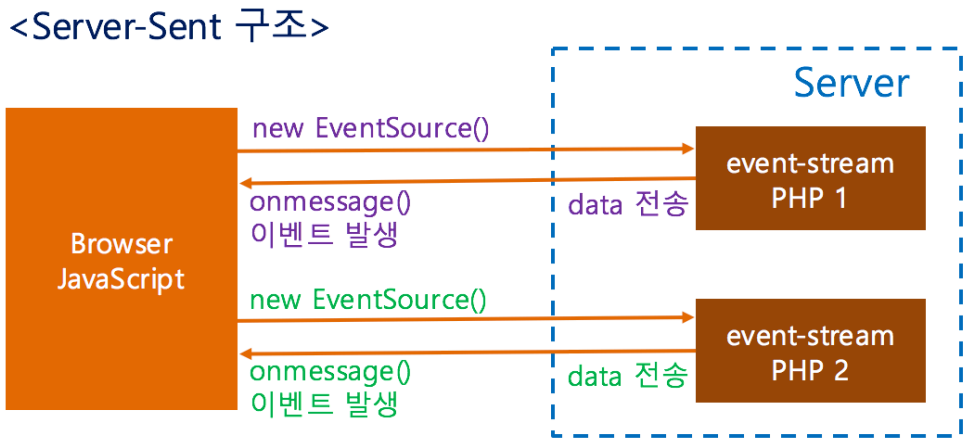
확인 사이트: http://jisiksoft.com/book/4/chapter_7/7.6/serverSent.html

그림 7-10 서버의 현재시간과 CPU/Memory 사용률을 실시간으로 받아서 보여준다.



[그림 7-11]은 'Server-Sent Events'의 동작 구조를 보여주는데, 브라우저는 EventSource() 생성자를 사용해서 서버의 PHP 프로그램을 실행하고 서버에서 데이터를 전송할 때마다 생성된 EventSource 클래스의 onmessage() 이벤트 함수가 자동으로 실행되고 서버로부터 받은 데이터를 처리한다.

그림 7-11 서버는 데이터를 계속해서 전송하고 브라우저는 onmessage 이벤트로 데이터를 받는다.



1. EventSource() 생성자로 'event-stream'을 전송하는 서버의 PHP 프로그램을 실행한다.
2. 서버의 PHP 프로그램은 몇초마다 주기적으로 data를 전송한다.
3. 서버에서 data를 전송할때마다 브라우저에서는 onmessage 이벤트가 발생한다.

[코드 7-16]은 화면에 보여지는 내용을 만드는 HTML 코드이며 'Start' 버튼을 클릭하면 startSent() 함수를 실행해서 서버로부터 실시간 데이터들을 받아서 화면에 출력하는 기능을 실행한다.

[코드 7-17]은 startSent() 함수를 구현한 JavaScript 코드이며 EventSource() 생성자를 사용해서 서버의 PHP 프로그램을 실행해서 실시간 데이터를 받는다.

[코드 7-16] 서버에서 보내는 정보를 받아서 화면에 보여주는 HTML 코드 (/js/serverSent.html)

```
<!DOCTYPE html>

<head>
	<title>Hello JisikSoft</title>
	<meta charset="utf-8"></meta>
	<link rel="stylesheet" href="./css/serverSent.css"></link>
	<script src="./js/jquery-2.1.3.min.js"></script>
	<script src="./js/serverSent.js"></script>
</head>
<body>
	<br>
```

```
    <div id='subject'>ServerSent Event of HTML5</div>
    <br><br>
    <div id='contents'>'ServerSent Event' is good to Update Real-Time Data of a server.</div>
    <div class='line2'></div>
    <div class='textLeft'>Server Time :  </div><div class='text' id='text1'>0</div>    //---①
    <div class='line1'></div>
    <div class='textLeft'>Server CPU :  </div><div class='text' id='text2'>0</div>
    <div class='line1'></div>
    <div class='textLeft'>Server Memory :</div><div class='text' id='text3'>0</div>
    <div class='line2'></div>
    <div class='button' onclick="startSent('text1', 'text2', 'text3');">Start</div>          //---②
</body>
</html>
```

① 'Server-Sent Events'를 사용해서 서버로부터 주기적으로 받는 메시지를 화면에 출력하기 위해서 'text'라는 클래스 값을 갖는 세 개의 태그들을 만들었다.

② 'Start' 버튼을 만들기 위해서 태그를 사용했으며, 클릭하면 startSent() 함수를 실행해서 'Server-Sent Events'를 실행한다.

[코드 7-17] 'Server-Sent Events'를 사용해서 실시간 정보를 받는 JavaScript 코드 (/js/serverSent.js)

```
var e1, e2, e3;                                                    //---①

function startSent(container1, container2, container3) {           //---②
    if (typeof(EventSource) !== "undefined") {                     //---③
        if(typeof(e1) == "undefined") {                            //---④
            e1 = new EventSource("./php/sent1.php");               //---⑤
            e1.onmessage = function(event) {                       //---⑥
                $('#'+container1).html(event.data);
            }
        }
        if(typeof(e2) == "undefined") {
            e2 = new EventSource("./php/sent2.php");
            e2.onmessage = function(event) {
                $('#'+container2).html(event.data);
            }
        }
        if(typeof(e3) == "undefined") {
            e3 = new EventSource("./php/sent3.php");
            e3.onmessage = function(event) {
                $('#'+container3).html(event.data);
            }
```

```
        }
    } else {
        alert("브라우저가 HTML5를 지원하지 않습니다.");
    }
}
```

① 'Server-Sent Events'를 수행하는 EventSource() 생성자를 사용해서 만드는 세 개의 EventSource 클래스들을 가리키기 위해서 세 개의 전역변수를 만들었다.

② 세 개의 EventSource 클래스들을 생성하고, 각각의 클래스의 onmessage 이벤트를 통해서 받은 메시지들을 화면에 출력하기 위해서 세 개의 매개변수들(container1, container2, container3)를 id 값으로 갖는 태그에 서버로부터 받은 메시지를 넣는다.

③ HTML5의 'Server-Sent Events' 기능을 지원하는 브라우저인지 확인하기 위해서는 EventSource 클래스가 정의되어 있는지를 확인한다. 만약, EventSource가 정의되지 않았다면 HTML5를 지원하지 않는 브라우저라는 경고 창을 발생시킨다.

④ EventSource() 생성자로 단 한 번만 클래스를 생성하는데, 만약 이전에 클래스가 생성되었다면 e1 변수는 값을 가지고 있기 때문에 추가생성을 하지 않는다.

⑤ EventSource() 생성자로 'Server-Sent Events'를 시작하는데, 생성자의 매개변수로 서버에서 실행하는 PHP 파일 이름을 넣는다. './php/sent1.php' 파일에서 구현한 코드는 몇 초마다 주기적으로 메시지를 전송한다.

⑥ 서버에서 전송하는 메시지가 있을 때마다 onmessage 이벤트가 실행되고, 전달받은 event를 통해서 'event.data' 변수에 있는 메시지를 화면에 출력한다.

[코드 7-18]부터 [코드 7-20]까지의 코드는 서버에서 실행되는 세 개의 PHP 프로그램들이며, 서버의 '현재시간', 'CPU 사용률', 'Memory 사용률'을 브라우저로 계속 전송한다. 서버의 PHP 프로그램은 브라우저와의 연결을 유지한 상태로 프로그램을 반복 실행하는데, 전송하는 데이터를 문자형식의 'event-stream'으로 설정해서 스트리밍 형식으로 만들고 'no-cache'로 설정해서 브라우저의 메모리에 데이터들이 쌓이지 않게 한다. 또한, 이전까지 사용했던 PHP 언어의 exit() 함수는 프로그램을 종료하기 때문에, exit() 함수 대신에 flush() 함수를 사용해서 데이터를 브라우저로 전송하고 프로그램을 종료하지 않는다.

[코드 7-18] 서버의 현재시간을 브라우저로 주기적으로 전송하는 PHP 코드 (/php/sent1.php)

```
<?php
```

```
header('Content-Type: text/event-stream');                    //---①
header('Cache-Control: no-cache');                            //---②

$time = date("h:i:sa");                                       //---③
echo "data: ".$time."\n\n";                                   //---④
flush();                                                      //---⑤
?>
```

① 브라우저에 데이터를 전달하기 전에 header() 함수를 사용해서 데이터의 속성을 정하는데, 데이터는 문자Text형식의 'event-stream'으로 계속 전달되는 데이터라고 브라우저에 알려준다.

② 모든 브라우저는 Cache(캐시)라는 저장공간을 가지고 있는데, 이전에 방문한 홈페이지에 다시 방문하면 홈페이지의 이미지 등을 서버로부터 받는 것이 아니라 이전에 Cache에 저장된 이미지를 사용한다. 이와 같이, Cache를 사용하면 이전에 방문한 사이트를 좀더 빨리 브라우저에서 보여줄 수 있다. 그러나, header() 함수를 사용해서 'no-cache'를 설정하면 브라우저에 전달하는 데이터를 Cache에 저장하지 않는다.

③ 서버의 현재시간을 문자열로 만들어서 $time 변수에 저장한다.

④ 브라우저로 보내는 데이터는 $time 변수에 있지만, 전달하는 문자열에는 항상 "data:"로 시작하고 "₩n₩n" 문자열을 마지막에 넣는다. 브라우저에서는 JSON 형식으로 인식해서 event.data 값으로 $time 변수의 값을 받아서 사용한다.

⑤ flush() 함수는 브라우저로 보내는 데이터를 전송하는 함수이며, 전송이 끝났다는 신호를 보내지 않는다. 즉, PHP 코드에서 exit() 함수를 사용하면 PHP 프로그램은 종료되는데, flush() 함수를 사용해서 연결이 종료되지 않고 'event-stream' 형식의 데이터를 계속 보내게 되며, 'no-cache'로 설정해서 브라우저의 Cache에 데이터가 쌓이지 않게 하면서 계속해서 데이터를 전송한다.

[코드 7-19] 서버의 CPU 사용률을 브라우저로 주기적으로 전송하는 PHP 코드 (/php/sent2.php)

```
<?php
header('Content-Type: text/event-stream');
header('Cache-Control: no-cache');

$load = sys_getloadavg();                                     //---①
echo "data: ".($load[0] * 100)." %\n\n";                      //---②
flush();
```

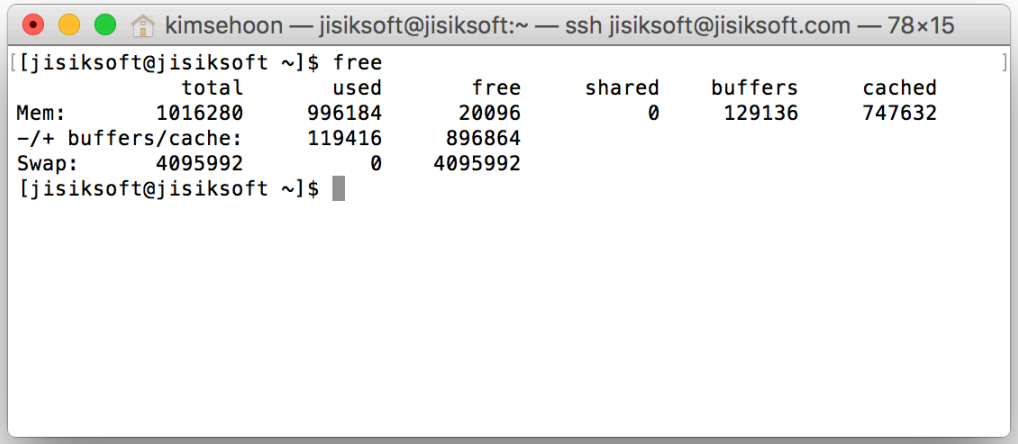
```
?>
```

① PHP 언어에서 제공하는 sys_getloadavg() 함수를 사용해서 CPU 사용률을 가져오는데, sys_getloadavg() 함수는 배열을 반환하고 배열에서 $load[0]에 저장된 값이 CPU 사용률을 나타낸다.

② 서버로 보내는 데이터를 "data:"와 "₩n₩n" 문자열을 사용해서 만드는데, "load[0] * 100" 연산을 수행해서 CPU 사용률을 백분율로 표시한다.

서버의 메모리 사용정보는 'free' 명령어로 확인하는데, 'free' 명령어의 결과를 가지고 메모리 사용량을 계산할 수 있다. [그림 7-12]는 서버에서 'free' 명령어를 실행한 결과를 보여주는데, 전체 메모리는 "total = 1016280(약 1-Gbytes)"이고 사용중인 메모리는 "used = 996184"이다. 즉, 메모리 사용률은 "used / total" 식으로 계산하면 된다.

그림 7-12 서버에서 'free' 명령어로 메모리 정보를 확인한 결과



[코드 7-20]은 메모리 사용률을 계산하기 위해서 [그림 7-12]에서 설명한 'free' 함수로 메모리 정보를 가져온다. PHP 언어에서 shell_exec() 함수는 리눅스의 Shell 함수를 실행한 결과를 반환하는데, shell_exec('free')를 실행해서 문자열로 결과값을 가져오고 문자열에서 메모리의 전체 사용량과 사용 중인 메모리 정보를 가져와서 메모리 사용량을 계산한다.

[코드 7-20] 서버의 Memory 사용률을 주기적으로 전송하는 PHP 코드(/php/sent3.php)

```
<?php
```

```
header('Content-Type: text/event-stream');
header('Cache-Control: no-cache');

$free = shell_exec('free');                                                    //---①
$free = (string)trim($free);
$free_arr = explode("\n", $free);
$mem = explode(" ", $free_arr[1]);                                             //---②
$mem = array_filter($mem);
$mem = array_merge($mem);
$memory_usage = $mem[2]/$mem[1]*100;                                           //---③
$result = substr($memory_usage, 0, strpos($memory_usage,'.'));                 //---④

echo "data: ".$result." %\n\n";                                                //---⑤
flush();
?>
```

① shell_exec() 함수를 사용해서 리눅스의 'free' 명령어를 실행해서 결과를 문자열로 반환한다. trim() 함수는 $free에 저장된 문자열에서 앞과 뒤에 존재하는 공백을 제거한다. explode() 함수를 사용해서 '₩n'(줄바꿈 문자)를 기준으로 문자열을 분리하고 배열로 만든다.

② 메모리 사용량 정보를 가지고 있는 두 번째 문자열($free_arr[1])을 공백을 기준으로 분리한다. 공백Space을 기준으로 분리된 배열은 데이터의 내용이 없는 항목들도 있기 때문에 array_filter() 함수를 사용해서 필요 없는 내용을 없애고 array_merge() 함수를 사용해서 내용을 가지고 있는 배열로 다시 만든다.

③ 사용 중인 메모리양(mem[2])을 전체 메모리양(mem[1])으로 나누고 100을 곱해서 '메모리 사용률'을 계산한다.

④ 메모리 사용률은 소수점 이하 값까지 나오기 때문에 문자열의 일부분을 가져오는 substr() 함수를 사용해서 정수값만을 갖는다.

⑤ 서버로 보내는 데이터를 "data:"와 "₩n₩n" 문자열을 사용해서 '메모리 사용률'을 포함한 문자열로 만들어서 전송한다.

인터넷을 사용하는 대부분의 사용자들은 그래프로 모니터링하는 화면을 접할 기회가 없지만, 최근에 만들어지는 많은 소프트웨어들은 관리자 화면을 브라우저에서 실시간으로 모니터링할 수 있게 구성하고 있다. 정적 페이지는 인터넷 상에서 접하는 대부분의 홈페이지들을 의미하고, 동적 페이지는 모니터링 화면과 같이 실시간으로 화면이 변하면서 사용자가 제어할 수 있게 만든 페이지를 의미한다. 고급 웹 프로그

래밍이란 동적 페이지를 만드는 작업을 말하는데, 기존의 프로그래밍 코드를 재사용하지 않고 JavaScript 언어로 모든 동작들을 구현하기 때문에 상당한 시간과 노력을 필요로 한다. 최근에 개발되는 동적 프로그램에서 HTML5의 기술을 많이 사용하고 있는데, Explorer 브라우저보다는 HTML5의 모든 기능을 충분히 지원하고 있는 Chrome이나 Firefox 브라우저에 최적화되게 만들고 있다.

부록

A.1 Firebug를 이용한 홈페이지 분석

HTML5와 같이 최근에 개발된 기능을 주로 다루는 웹 프로그래머들는 Firefox와 Chrome 브리우저를 좋아하는데, Firefox는 디버깅을 하는데 편리한 기능이 많고 Chrome은 상당히 빠르게 동작한다. 웹 프로그램을 만들어서 서비스를 오픈하게 되면 사용자들은 다양한 브라우저를 사용하기 때문에 서비스를 오픈하기 전에 모든 브라우저에서 정상적으로 동작하는지를 확인해야 하는데 웹 프로그래머에게는 다소 귀찮은 작업일 수 있다. 예를 들면, 브라우저마다 픽셀값을 맞춰야 하는 작업으로 많은 시간을 할애하는 경우도 발생한다. 웹 프로그래머는 컴퓨터 운영체제에 종속되지 않는 서버스를 제공해야 하지만, 모든 브라우저에서 문제 없이 동작하는 것을 확인해야 하는 최종 작업을 항상 진행해야 한다. HTML은 더 많은 기능을 추가하면서 계속 발전하는데, 브라우저들도 HTML의 모든 기능을 처리할수 있게 발전하고 있다. 우리나라에서 가장 많이 사용하고 있는 Explorer 브라우저를 웹 프로그래머들은 좋아하지 않는 편인데, 다른 브라우저에 비해서 HTML의 새로운 기능들을 처리할 수 있게 발전하는 속도가 늦기 때문이다. 또한 현재도 버전이 낮은 Explorer를 많이 사용하고 있는 우리나라에서는 HTML5를 사용해서 웹 페이지를 만들어도 사용자들이 활용할 수 없는 경우가 많다. 웹 프로그래밍을 하면서 다수의 브라우저들을 사용해 본 필자는 개인적으로 Firefox 브라우저를 좋아하는데, Firefox 브라우저에서 사용하는 Firebug 툴이 디버깅을 하는데 상당히 편리하기 때문이다. 브라우저에서 처리하는 HTML, CSS, JavaScript 파일들은 Firebug에서 모든 코드를 확인할 수 있고 필요에 따라서 설정값들을 변경할 수도 있다. 다른 사람이 만든 프로그램의 코드를 확인할 수 있는 웹 프로그래밍에서는 Firebug와 같은 툴을 사용해서 다른 곳에서 적용된 코드를 쉽게 복사해서 사용할수 있으며, 실제 많은 웹 프로그래머들이 이를 활용하고 있다. 이 책에서 설명한 웹 프로그래밍의 기본 원리를 이해한 독자들은

Firebug를 사용해서 디버깅을 하는 방법을 익히고 나면 인터넷에서 접근 가능한 대부분의 웹 페이지의 코드를 분석할 수 있다. 물론, 파일 크기를 작게 하기 위해서 인코딩 작업을 진행한 후에 배포하는 JavaScript 파일들의 코드를 확인하는 것은 힘들지만, 비용을 받고 제공하는 파일이 아닌 경우에는 오픈 소스 개념으로 원본Source Code 파일들을 같이 제공하는 경우가 많다. Chrome 브라우저의 '개발자 도구'도 Firefox의 Firebug와 같은 기능을 제공하지만, Firebug가 좀더 사용하기 편리하기 때문에 이 책에서는 Firebug만을 설명한다. Firebug는 브라우저에 Firefox 브라우저에 설치해야하는 개별 프로그램인데, Firebug 홈페이지에서 다운받아 설치하면 된다. (https://getfirebug.com/downloads)

[그림 A-1]은 Firebug가 설치된 Firefox 브라우저를 보여주는데, 브라우저의 상단에 '개똥벌레firebug 이미지'가 추가되었고 이것을 클릭하면 오른쪽 그림과 같이 아래부분에 Firebug가 활성화된다. Firebug는 브라우저에서 다운받은 모든 파일 내용을 확인할 수 있으며, 특정 영역에서 에러가 발생하면 에러가 발생한 코드의 위치를 알려준다. Firefox에서 가장 많이 사용하는 기능은 마우스로 브라우저 화면의 특정 영역의 코드를 확인하는 것이며, [그림 A-2]는 마우스를 브라우저의 특정 부분으로 이동한 후에 해당 영역의 HTML 코드와 디자인 속성을 Firebug 창에서 확인하는 그림을 보여준다. Firebug 창에서 HTML 코드의 속성값이나 CSS 디자인 속성값의 변경도 가능한데, 만약 오른쪽의 디장인 속성값을 변경하면 브라우저에 즉시 적용되어 변경된 디자인을 확인할 수 있다.

그림 A-1 Firefox 브라우저에 설치된 Firebug 를 실행한 화면

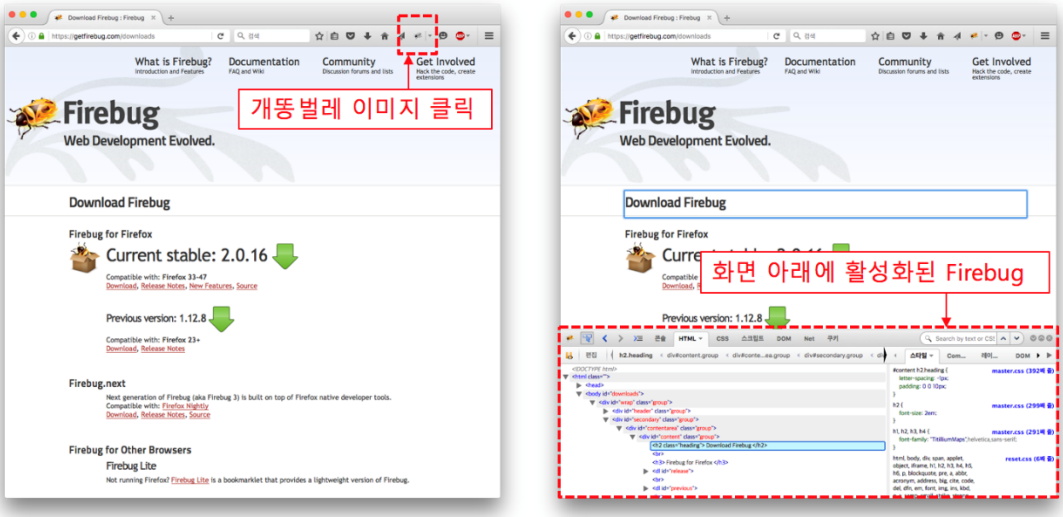
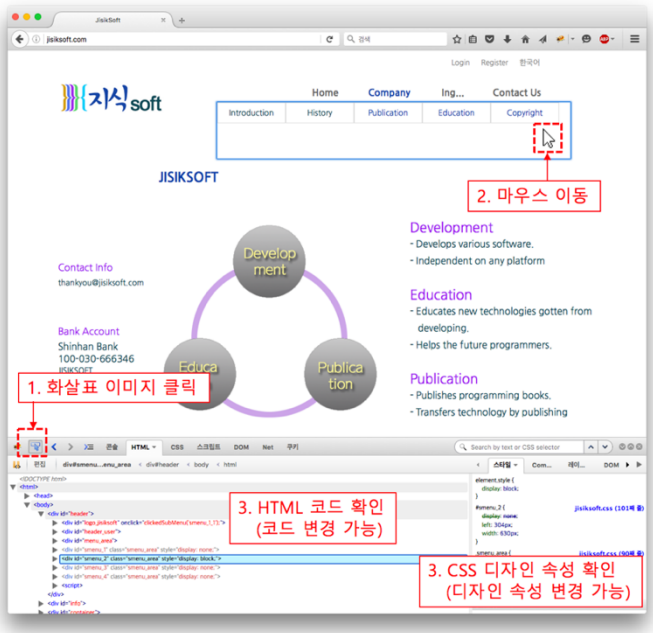


그림 A-2 마우스로 특정 영역의 HTML과 CSS 디자인 속성을 확인하고 값을 변경할 수 있다.



Firebug는 브라우저에서 다운받은 모든 파일의 내용을 확인할 수 있는데, JavaScript 파일들을 보기 위해서는 '스크립트' 탭을 클릭하고 확인하려는 JavaScript 파일을 선택하면 된다. [그림 A-3]은 '스크립트' 탭을 사용해서 다운받은 모든 JavaScript 파일들의 리스트를 보는 방법을 보여준다. [그림 A-4]는 JavaScript 코드 화면의 왼쪽에 마우스를 클릭해서 Breakpoint(중단점)을 만들어서 코드를 단계적으로 수행하면서 디버깅하는 화면을 보여주는데, 대부분의 프로그래밍 개발 환경에서 제공하는 Breakpoint를 Firebug에서 제공한다.

그림 A-3 현재 페이지에서 다운받은 JavaScript 파일의 내용을 확인할 수 있다.

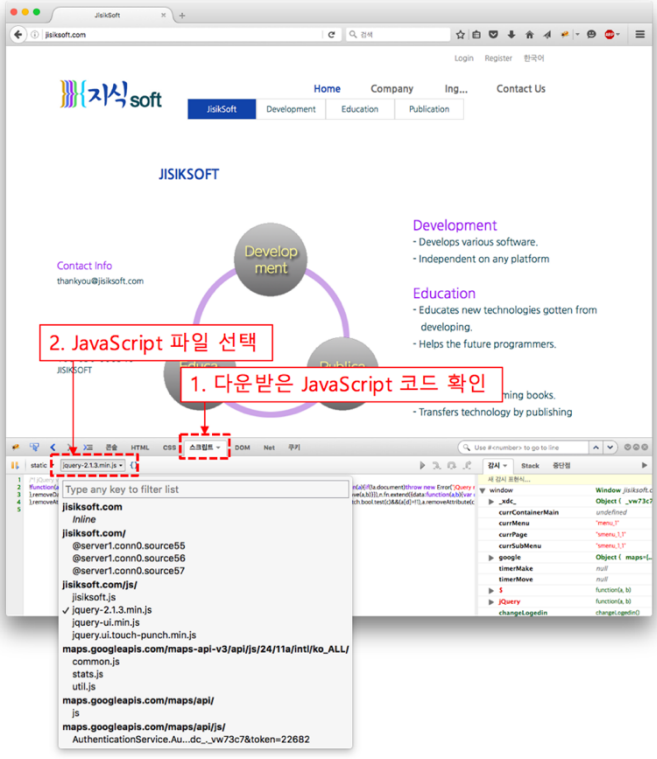
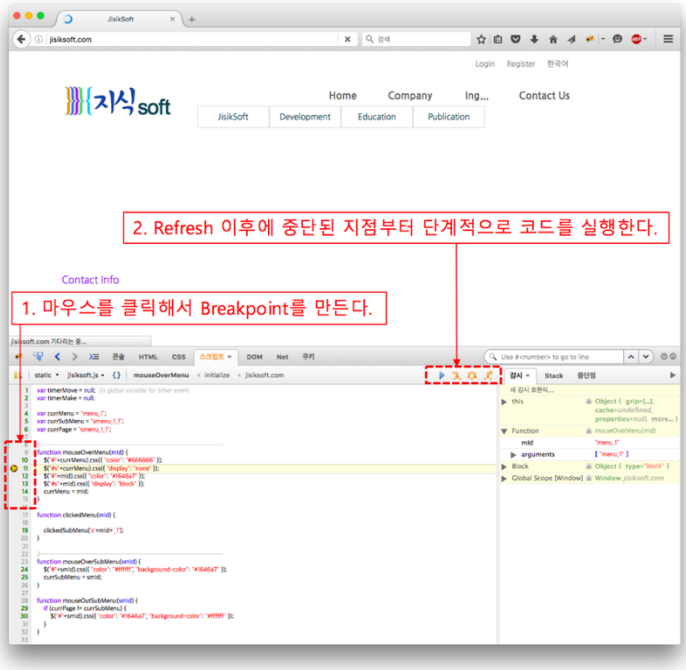
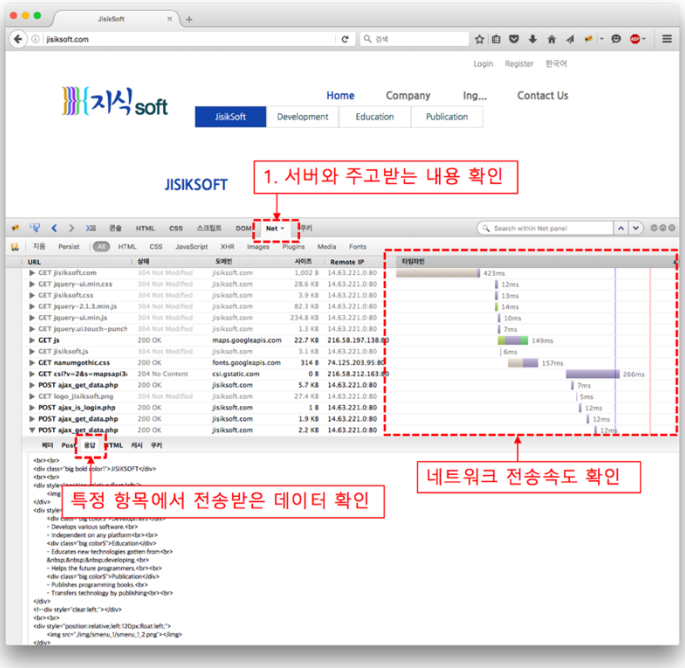


그림 A-4 Breakpoint 를 사용해서 JavaScript 코드를 디버깅하는 방법



웹 프로그램은 개발하면서 네트워크 상에서 주고받는 데이터들을 확인하는 것이 중요한데, 'Net' 탭을 클릭하면 서버로부터 받는 데이터의 내용과 전송속도를 확인할 수 있다. 우리나라의 인터넷 망은 속도가 빠르기 때문에 특정 데이터를 받는데 속도가 길어진다면 해당 데이터를 보내는 서버의 문제일 가능성이 높다. 사진과 같은 용량이 큰 데이터의 전송속도는 다른 데이터에 비해 긴 편인데, 많은 사용자가 접속하는 서비스를 개발할 때는 전송속도를 측정해서 시스템의 문제점을 파악하기도 한다. [그림 A-5]는 'Net' 탭을 클릭해서 전송속도와 전달되는 데이터의 내용을 확인하는 방법을 보여준다.

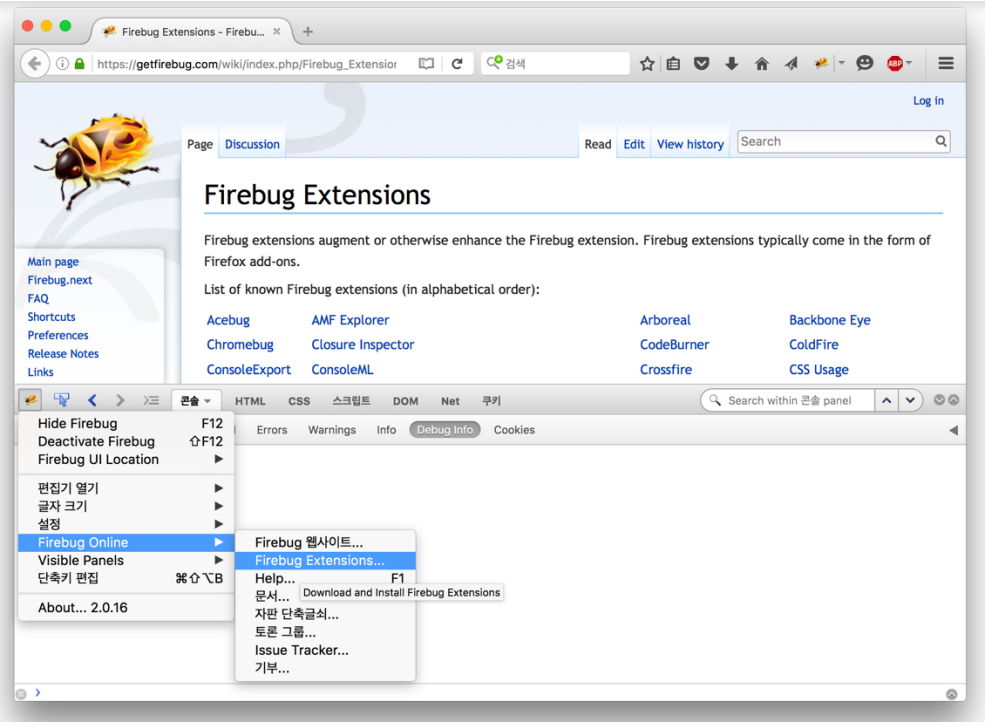
그림 A-5 네트워크의 전송속도와 데이터 확인



Firebug에서는 브라우저의 객체들의 내용을 확인할 수 있는 'DOM' 탭이 있는데, 웹 프로그래밍을 하면서 'DOM' 구조까지 확인할 일은 거의 없기 때문에 여기서는 생략한다. 웹 프로그램을 만들면서 '쿠키'라는 단어를 종종 접하는데, '쿠키'는 서비스를 제공하는 서버에서 브라우저에 특정 데이터를 저장하기 위해서 사용하는 기능이다. 예를 들면, 쿠키에 암호화된 값을 저장하고 해당 브라우저가 정상적으로 서버와 통신을 하는지를 확인하기 위해서 쿠키를 사용하기도 한다. 웹 프로그래밍을 하면서 '쿠키'에 저장하는 내용이 있으면 '쿠키' 탭에서 쿠키의 이름과 값을 확인할 수 있다. 만약 쿠키를 사용해서 사용자의 브라우저에 저장할 내용이 있으면 항상 암호화된 내용을 사용하는 것이 좋은데, Firefox에는 쿠키의 값을 변경하는 기능도 있기 때문이다. 간단한 웹 프로그램을 만들 때 대부분의 방문자가 확인할 수 없는 쿠키의 사용은 특별한 경우가 아니면 사용하지 않는 것이 좋은데, 만약 악성 프로그램이 브라우저에 설치되었고 쿠키에 개인정보가 저장되었을 경우에 악성 프로그램이 쿠키의 정보를 가져갈 수도 있기 때문이다.

[그림 A-6]은 Firebug의 기본기능 외에 확장된 기능들을 추가설치하는 방법을 보여주는데, firebug 창의 왼쪽 위에 있는 '개똥벌레 이미지'를 클릭한 후 'Firebug Online'/'Firebug Extensions'을 선택하면 확장된 기능들을 설치하는 페이지로 이동한다.

그림 A-6 Firebug 확장기능들을 추가 설치할 수 있다.



오래전 HTML5를 사용해서 웹 페이지를 만들 때 Firefox와 Chrome 브라우저로 오랫동안 테스트를 진행했었다. 브라우저의 처리속도 측면에서는 Chrome 브라우저가 성능이 좋았지만, 장시간 데이터를 받으면서 많은 연산을 수행할 때는 Firefox가 좀 더 안정적이었다. 우리나라에서는 '윈도우' 운영체제를 사용하기 때문에 세 개의 브라우저들(Explorer, Chrome, Firefox) 중의 하나를 선택해야 하는데, 특정 사용자가 다루는 웹 페이지를 HTML5 기능들을 사용해서 만든다면 Chrome이나 Firefox를 기준으로 개발하는 것이 좋다.

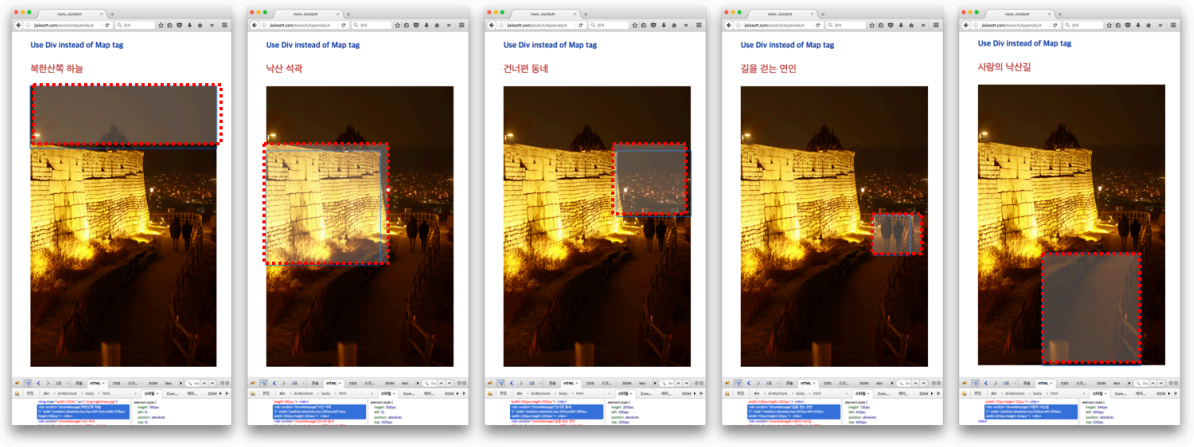
A.2 <map> 태그를 대체하는 태그

1990년대 말에 브라우저를 사용해서 인터넷을 사용하게 되었는데, 초창기에는 간단한 HTML 태그를 사용해서 글자와 사진 등의 정보를 볼 수 있었다. 필자의 기억으로는 2000년대 초에 <map></map> 태그를 사용해서 조금더 화려한 기능을 제공하는 홈페이지들이 만들어졌는데, 사진의 특정 영역으로 마우스를 이동하면 다양한 효과를 줄 수 있었다. <map></map> 태그는 사진의 특정 영역에 범위를 정하고 해당 영역에서 발생하는 마우스 이벤트를 처리하는데, <map></map> 태그에서 처리하는 방법은 태그를 사용해서 간단히 대체할 수 있다. 이번 장에서는 태그를 사용해서 사진의 특정 영역을 클릭했을 때 이벤트를 발생하는 예제를 보여주는데, HTML에서 제공하는 모든 종류의 태그들을 기억하는 것보다는 태그와 태그 안에서 발생하는 이벤트를 적절히 사용하면 다른 태그들을 쉽게 대체할 수 있다.

[그림 A-7]는 사진에서 특정 영역을 마우스로 클릭하면 해당 영역에 대한 설명을 화면 위에 출력하는 결과를 보여준다. 눈으로 확인을 할 수는 없지만, 사진에서 태그들을 사용해서 5개의 영역을 만들고 onclick 이벤트 처리를 하였다.

확인 사이트: http://jisiksoft.com/book/4/Appendix/B/mapDiv.html

그림 A-7 태그를 사용해서 사진의 특정 부분에 대한 이벤트를 처리한다.



<map></map> 태그에서 영역을 정할때 픽셀Pixel 단위로 범위를 정하는데, 태그도 마찬가지로 위치와 범위를 픽셀단위로 설정한다. [코드 A-1]은 사진을 브라우저에서 보여주기 위해서 'picture'라는 태그를 만들었으며, 이 태그 안에 <img> 태그를 사용해서 사진을 불러오고 5개의 태그들을 사용해서 사진 안에서의 영역을 설정했다. [그림 A-7]에서와 같이 Firebug 툴을 사용하면 태그의 영역을 확인하는데 편리하다. 영역을 만드는 다섯 개의 태그에는 onclick 이벤트를 사용해서 showMessage() 함수를 실행하고, showMessage() 함수는 각각의 영역에 대한 설명을 화면에 출력한다.

[코드 A-1] 태그로 사진의 이벤트를 처리하는 코드 (mapDiv.html)

```
<!DOCTYPE html>

<head>
	<title>Hello JisikSoft</title>
	<meta charset="utf-8"></meta>
	<link rel="stylesheet" href="./css/mapDiv.css"></link>
	<script src="./js/jquery-2.1.3.min.js"></script>
</head>
<body>
	<br>
	<div id='subject'>Use Div instead of Map tag</div>
	<br><br>
	<div id='message'></div>
	<br><br>
	<div id='picture' style="position:relative;left:50px;width:576px;height:864px;">
	   <img src="./img/nightView.jpg" style="width:100%;">                    //---①
	   <div style="position:absolute;top:0px;left:0px;width:576px;height:190px;"
	                  onclick="showMessage('북한산쪽 하늘');"></div>          //---②
	   <div style="position:absolute;top:200px;left:0px;width:350px;height:350px;"
```

```
                    onclick="showMessage('낙산 석곽');"></div>
        <div style="position:absolute;top:200px;left:350px;width:226px;height:200px;"
                    onclick="showMessage('건너편 동네');"></div>
        <div style="position:absolute;top:400px;left:410px;width:110px;height:120px;"
                    onclick="showMessage('길을 걷는 연인');"></div>
        <div style="position:absolute;top:520px;left:200px;width:300px;height:344px;"
                    onclick="showMessage('사랑의 낙산길');"></div>
    </div>
    <br><br>

     <script>

        $(window).bind("mousedown contextmenu", function() {
                return false;
        });

        function showMessage(msg) {                                          //---③
          $('#message').html(msg);
        }

     </script>

</body>
</html>
```

① 'picture'라는 id 값을 가진 태그에 사진을 출력하기 위해서 <img> 태그를 사용해서 사진을 불러온다. 전체 영역에 사진을 출력하기 위해서 width 속성을 100%로 설정했다.

② 사진의 특정 영역에서의 이벤트 처리를 위해서 추가된 다섯 개의 태그들은 'position' 속성을 'absolute'로 설정해서 사진의 영역에 종속되어 위치와 영역의 크기를 픽셀 단위로 계산한다. 태그 안에서는 onclick 이벤트를 사용해서 showMessage() 함수를 실행한다.

③ showMessge() 함수는 매개변수로 전달받은 msg 변수의 내용을 'message'라는 id 값을 가진 태그 안에 넣는다.

HTML 언어의 표준은 계속 발전하고 있으며, 현재는 <map></map> 태그를 많이 사용하지 않는다. 또한 웹 프로그래밍의 코드를 직접 구현하는 프로그래머들은 깊이 알고 있는 몇가지의 내용을 응용해서 새로운 것으로 발전시키려 노력한다. 예를 들면, 태그만으로도 많은 것들을 구현할 수 있기 때문에 종류가 많은 HTML 태그들의 특성을 모두 기억하고 프로그래밍하기 보다는 기본으로 사용하는 태그의 활용을 연구하는 것도 좋은 방법이다. 즉, 모든 태그에는 모든

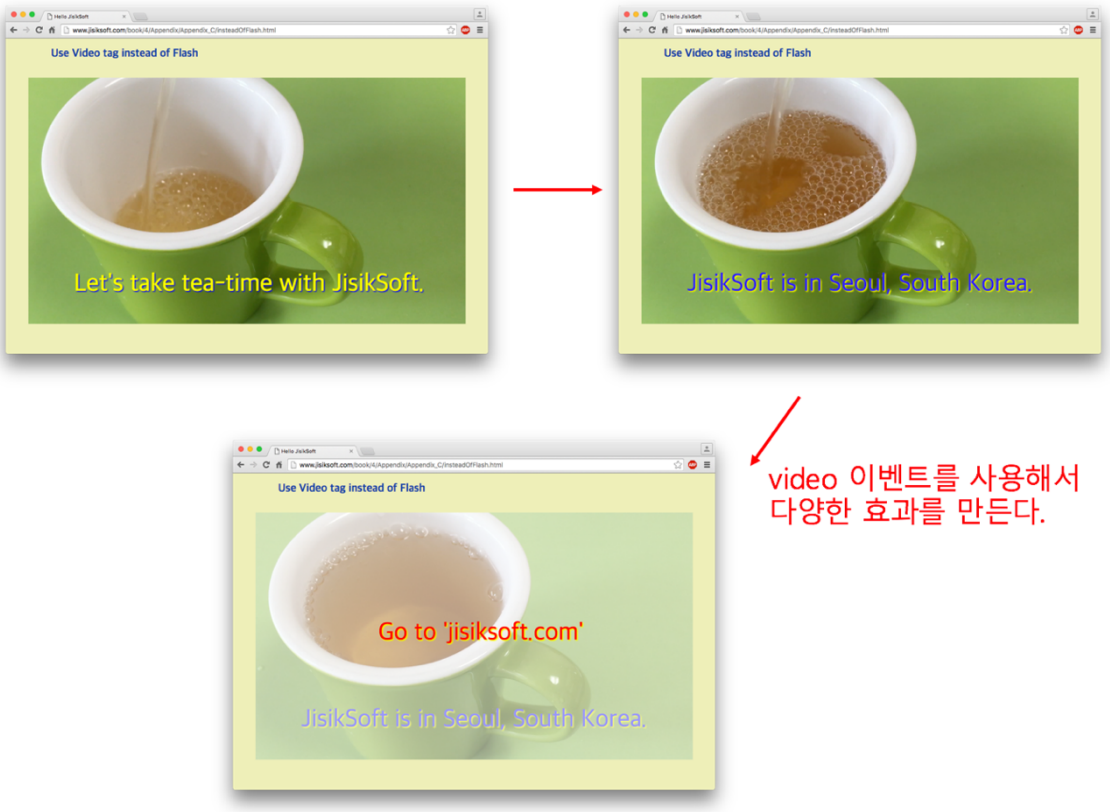
이벤트를 사용할 수 있는데, 태그만을 사용해서 모든 이벤트를 처리하는 코드를 만드는 것도 필자는 추천한다.

A.3 Flash 대신에 <video> 태그를 사용한다.

2000년대에는 Flash를 사용해서 화려한 홈페이지를 만드는 것이 유행이었는데, 화장품 회사나 자동차 회사의 홈페이지들은 고급스러운 이미지를 표현해야 하기 위해서 Flash를 많이 사용했다. 2010년대부터는 전문가들이 Flash가 점차적으로 없어질 것이라고 하고, 또한 Flash를 기본적으로 지원하지 않는 브라우저도 있었지만 현재도 국내의 사이트들은 Flash를 사용하는 곳이 많다. 빠른 통신망을 사용하는 국내에서는 Flash의 용량이 크더라도 사이트를 이용하는데 큰 무리가 없기 때문이다. 그러나, HTML5의 <video></video> 태그를 잘 활용하면 Flash 기능을 사용하지 않아도 고급스러운 홈페이지를 만들 수 있다.

확인 사이트: http://jisiksoft.com/book/4/Appendix/C/insteadOfFlash.html

그림 A-8 <video> 태그에서 발생하는 이벤트를 사용해서 다양한 효과를 만든다.



[그림 A-8]은 <video></video> 태그와 Javascript 언어를 사용해서 동영상에서 글자를 출력하고 이벤트 처리를 받는 화면을 구현하였다. 이전 부록에서 사진에 태그로 영역을 만들어서 이벤트를 처리했는데, <video></video> 태그에서도 영상이 나오는 영역에 태그들을 사용해서 이벤트를 처리할 수 있다. <video></video> 태그는 고유의 이벤트들을 가지고 있는데, 영상이 시작하고 종료할 때 발생하는 이벤트와 영상이 진행되는 동안에 특정시간에 이벤트를 처리하는 기능이 많이 사용되고 있으며, 이외에도 다양한 이벤트 기능들을 제공한다. 이 책에서는 영상이 끝난 이후에 발생하는 'ended' 이벤트와 영상이 시작되고 특정 시간에 Javascript 함수를 실행하기 위해서 'timeupdate' 이벤트를 사용했다. 'ended' 이벤트는 동영상이 끝나는 시점에 한번만 발생하지만, 'timeupdate' 이벤트는 동영상이 진행하는 동안 계속해서 발생한다. 'timeupdate' 이벤트를 사용할 때는 동영상이 진행되는 시간을 저장하고 있는 'currentTime' 변수를 사용해서 특정 시간에 동작을 수행하는 Javascript 함수를 실행한다. 이 책에서 구현하는 예제는 동영상이 실행하고 2초가 되면 메시지가 나타나고 10초에는 해당 메시지가 사라지고 12초에 다른 메시지를 출력한다. 동영상이 끝난 이후에는 동영상을 블라인드Blind로 처리해서 비활성화된 상태로 만들고, 다른 페이지로 이동할 수 있는 이벤트 메시지를 보여준다. [코드 A-2]은 <video></video> 태그에서 제공하는 이벤트를 사용해서 다양한 효과를 만드는 코드이며, 필자가 핸드폰으로 촬영한 동영상이 대신에 전문가가 제작한 동영상을 사용한다면 Flash보다 멋진 효과를 보여주는 사이트의 제작이 가능하다.

[코드 A-2] <video></video> 태그에서 제공하는 이벤트 사용 (insteadOfFlash.html)

```
<!DOCTYPE html>

<head>
    <title>Hello JisikSoft</title>
    <meta charset="utf-8"></meta>
    <link rel="stylesheet" href="./css/insteadOfFlash.css"></link>
    <script src="./js/jquery-2.1.3.min.js"></script>
</head>
<body>
    <br>
    <div id='subject'>Use Video tag instead of Flash</div>
    <br><br>
    <div style="position:relative;width:960px;height:540px;">                    //---①
      <video id="myVideo" style="width:100%;height:100%;" autoplay muted>   //---②
```

```
            <source src="./video/teaCup.m4v" type="video/mp4">
        </video>

        <div id="text1">Let's take tea-time with JisikSoft.</div>                    //---③
        <div id="text2">JisikSoft is in Seoul, South Korea.</div>

        <div id="blind"></div>                                                       //---④
        <div id="goto" onclick="location.href='http://jisiksoft.com'">
            Go to 'jisiksoft.com'</div>                                              //---⑤
    </div>
    <br><br>

     <script>

        $(window).bind("mousedown contextmenu", function() {
            return false;
        });

        var isFired_1 = false;                                                       //---⑥
        var isFired_2 = false;
        var isFired_3 = false;

        $('#myVideo').on('timeupdate', function() {                                  //---⑦
            if (!isFired_1 && $('#myVideo')[0].currentTime > 2) {
                $('#text1').animate({opacity:1}, 2000);
                isFired_1 = true;
            }
            if (!isFired_2 && $('#myVideo')[0].currentTime > 10) {
                $('#text1').animate({opacity:0}, 2000);
                isFired_2 = true;
            }
            if (!isFired_3 && $('#myVideo')[0].currentTime > 12) {
                $('#text2').animate({opacity:1}, 2000);
                isFired_3 = true;
            }
        });

        $('#myVideo').on('ended', function() {                                       //---⑧
            $('#blind').css({'display':'block'});
            $('#goto').css({'display':'block'});
        });

    </script>

</body>
</html>
```

① 동영상과 동영상에 보여지는 메시지들을 포함하는 태그로서 동영

상의 크기를 정하기 위해서 가로 960px과 세로 540px로 설정했다.

② 'myVideo'라는 id를 가진 <video></video> 태그이며 자동으로 실행하고 소리가 안나게 하기 위해서 autoplay와 muted로 설정했다. <video></video> 태그 안에 <source> 태그를 사용해서 'teaCup.m4v' 동영상을 화면에 보여준다.

③ 동영상의 아래에 보여지는 메시지들을 태그를 사용해서 만들었다. 처음에는 투명도opacity를 0으로 설정해서 화면에 보이지 않지만, 동영상이 실행되면 특정 시간에 투명도를 조정해서 글자를 동영상에서 보이게 한다.

④ 동영상이 끝난 후 비활성화 상태로 보이기 위해서 동영상 앞에 투명도가 있는 하얀색 블라인드를 만들기 위해서 'blind'라는 id 값을 가진 태그를 사용했다.

⑤ 동영상이 끝난 후 'jisiksoft.com' 홈페이지로 이동할 수 있는 링크Link를 화면에 출력한다.

⑥ 동영상이 실행되면서 발생하는 이벤트들을 위해서 세 개의 변수들(isFired_1, isFired_2, isFired_3)을 사용한다. 처음에는 false로 설정되지만, 이벤트가 발생할 때마다 하나씩 true로 변경된다.

⑦ 'myVideo'라는 id를 가진 <video></video> 태그의 'timeupdate' 이벤트를 처리한다. $('#myVideo')는 배열을 반환하며, 여기서는 <video></video> 태그가 하나이기 때문에 $('#myVideo')[0]으로 첫 번째 배열에 접근하고 해당 <video></video> 태그의 실행 시간을 저장하고 있는 currentTime를 사용해서 특정시간에 이벤트를 처리한다. 여기서는 'timeupdate' 이벤트가 계속해서 발생하는데, 2초가 지나면 'text1'라는 id를 가진 태그의 투명도를 단계적으로 높여서 화면에 보이게 하고 10초가 지나면 투명도를 다시 0으로 만들어서 화면에서 사라지게 하고, 12초가 되면 'text2'라는 id를 가진 태그의 투명도를 높인다.

⑧ 동영상이 종료되면 'ended' 이벤트가 발생하고, 'blind'와 'goto'라는 id를 가진 태그들을 화면에 보이게 한다.

HTML5에는 좋은 기능들이 많이 추가되었지만, 아직까지는 HTML5를 사용해서 사이트를 구축하는 곳은 많지 않다. HTML5의 기능 중 하나인 <video></video> 태그에서도 다양한 이벤트들을 제공하지만, 대부분의 개발자들은 이벤트를 사용할 필요성을 많이 느끼지는 않는다. 간혹, 광고를 통해서 수익을 얻는 곳에서 <video></video> 태그의 이벤트를 사용해서 광고를 사라지게 하는 버튼을 일정시

간 이후에 화면에 보이게 만드는데 사용하고 있다. 웹 프로그래밍을 하면서 HTML, CSS를 이해하고 이벤트들을 적절히 사용해서 JavaScript로 다양한 동작을 수행하게 만드는 기본적인 흐름을 이해하면 좀더 재미있는 사이트를 쉽게 만들수 있을 것이다.

글을 마치며

웹 프로그래밍에 대한 내용은 워낙 방대해서 HTML, CSS, JavaScript, jQuery, PHP, HTML5 등을 자세히 설명하면 각각의 주제를 설명는 것만으로도 책 한권의 분량이 나온다. 책을 집필하는 초기에는 웹 프로그래밍에서 사용하는 프로그래밍 언어들을 왜 사용하는지에 대한 이해만 하면 쉬울 것이라 생각했다. 작은 회사의 홈페이지가 아니고 많은 사용자가 접속하는 서비스를 운영하기 위해서는 개별 프로그래밍 언어에 해박한 지식을 가지고 있는 다수의 전문가들이 모여서 하나의 완벽한 시스템을 구축하고 있다. 현재 IT 시장에 처음 발을 들여놓는 사람들은 다양한 역할 중에서 웹 프로그래밍을 맡는 경우가 많은데 웹 프로그래밍에 대한 전반적인 내용을 이해하는데 이 책이 도움이 되었기를 바란다. 필자가 만들고 있는 책들은 초보자들이 주로 보는 입문서와는 다른 내용으로 프로그래밍 언어의 문법을 자세히 설명하지는 않는다. 하지만, 하나의 프로그래밍 언어를 이해한 독자라면 모든 프로그래밍 언어는 공통된 연산자들을 사용하기 때문에 이 책의 코드를 이해하는데 어렵지 않을 것이라 생각한다. 회사의 홈페이지를 직접 만드는데는 많은 시간이 소요되는데, 기존의 홈페이지 제작사들은 기본적인 골격을 가진 홈페이지를 처음에 구축하고 이후에 의뢰받는 홈페이지들은 이미지만을 변경해서 만드는 경우가 많다. 또한 홈페이지를 간편하게 제작할 수 있는 소프트웨어들도 시중에는 다수가 존재한다. 그러나, 이 책을 이해하고 간단한 웹 페이지를 구축해보면 웹 프로그래밍의 전반적인 구조를 이해 할 수 있으며, 이후에 대형 프로젝트에 참여해도 개발하는 시스템의 전체를 이해하고 자신이 맡은 분야가 무엇을 위한 것인지를 쉽게 파악할 수 있을 것이다.

이 책은 필자가 초안을 작성하는데만 6개월이 소요되었다. 책에서 구현한 모든 코드는 책의 내용에 맞추어서 필요한 프로그램들을 만든 것이고, 복잡한 코드도 있지만 대부분은 일반적인 웹 프로그램의 코드보다 간단한 구조를 갖는다. 프로그래밍을 공부하는 많은 사람들을 만나면서 느낀 것 중의 하나는 단시간 내에 많은 것을 얻어서 프로그래밍을 잘하고 싶어하는 사람들이 많다는 것이다. 그러나, 필자의 경험

상 프로그래밍을 몸으로 익히는 과정은 상당한 시간이 필요한데, 입문하는 사람들은 기초 서적을 읽고 나서 프로그래밍을 할 수 있게 되었다고 생각하는 경우가 많다. 그러나, 입문서는 프로그래밍의 기초단계일 뿐 스스로 하나의 프로그램을 만드는데는 한계가 있으며 많은 사람들이 실무에서 프로그래밍 실력을 향상시키는 과정을 거친다. 대학교에서 컴퓨터 관련 학과에서 공부한 사람들도 프로그래밍에 한계를 느끼고 다른 분야에서 일을 하는 경우가 많은데, 프로그래밍을 하면서 계속적으로 부딪히는 문제들을 풀어나가는데 지치는 경우도 발생하기 때문이다. 또한, 프로그램을 만들면서 인터넷에 있는 많은 정보들을 찾는 것은 잘하지만, 그 정보를 자신의 것으로 만드는 과정 없이, 이해하지 못한 코드를 적용하다보면 항상 새로운 것을 만들어야하는 프로그래밍의 세계에서 한계에 부딪히는 일을 경험하곤 한다. 필자의 만드는 'How-to Series'는 입문서적은 아니지만, 각각의 주제에서 구현하는 과정들을 이론적인 설명보다는 깔끔한 코드로 이해하는데 초점을 두었다. 충분한 시간을 가지고 독자도 자신만의 새로운 코드로 응용할 수 있기를 바란다.

학창시절부터 국어를 못했던 필자의 거친 초안을 편집해 준 나의 아내 '남혜정'에게 감사 드린다.