쌓임 규칙에 기반한 복합기능을 가진 풀다운메뉴 설계 방법

(A New Design Method for Multi-functional Pull-down menu based on Stacking Rules)

조한수*

(Cho Han Soo)

요 약

본 논문은 쌓임 규칙에 기반한 복합기능을 가진 풀다운메뉴의 새로운 설계 방법을 제안한다. 기존의 웹 내비게이션 관련 연구에서는 디자인적인 측면을 강조한 내용이 대부분이다. 본 논문은 프로그래밍기법에 주안점을 둔 기술적인 측면을 고려하여 다수의 엘리먼트로 풀다운메뉴의 메인메뉴를 구성하고, 이들을 쌓임 규칙을 이용하여 서로 다른 레이어에 배치함으로써 메인메뉴의 기본적인 분류기능 이외에 독립적으로 고유의 기능을 할 수 있다. 또한 본 논문에서 제안한 애니메이션기법을 사용한 서브메뉴 선택 바와 서브메뉴 항목 자동 탐색 기능을 추가하여 시각적인 효과와 정보탐색 효율을 향상할 수 있다. 본 논문에서 제안한 복합기능을 갖는 풀다운메뉴를 CSS와 jQuery를 사용하여 구현함으로써 제안한 방법의 유효성을 확인하였다.

■ 중심어 : 풀다운메뉴 ; 쌓임 규칙 ; 서브메뉴 선택 바

Abstract

A new design method for multi-functional pull-down menu based on stacking rules is proposed. The importance of a design point of view has been emphasized in many previous studies on web navigation. Taking technical aspects with an emphasis on programming techniques into consideration in this paper, multiple elements, which are used for constructing main menu of the pull-down menu, are capable of performing their unique functions independently in addition to basic classification function by disposing them to separated layers using stacking rules. Furthermore, the improvement of visual effects and efficiency for information navigation can be expected by implementing submenu selection-bar using animation techniques and function to automatically search submenu item. Finally, the effectiveness of the proposed method is identified by implementing multi-functional pull-down menu using CSS and jQuery.

■ keywords : pull-down menu ; stacking rules ; submenu selection-bar

Ⅰ. 서 론

웹 내비게이션 메뉴는 링크의 역할[1] 뿐만 아니라 웹 사이트에서 제공하는 정보의 전체 구조를 결정한다. 또한 사용자가 웹 사이트를 방문했을 때 가장 먼저 관심을 가지는 부분[2]이어서 웹 사이트를 구축하는데 중요한 역할을 한다. 웹 내비게이션과 관련된 지금까지의 연구로서는 메뉴디자인이 웹 사이트에 미치는 영향에 관한 연구[3-7], 메뉴구조에 대한 탐색효과와 관련된 연구[8] 등의 다양한 연구가 진행되어 왔다. 이러한 연구들은 주로 웹 내비게이션과 웹 사이트의 구조를 연관시키는 연구로서 실제로 웹 내비게이션을 제작하는 기법에 관한 연구는 거의 보고된 바가 없다. 최근에는 메뉴제작과 편집이 가능한 도구를 사용하여 제작된 웹 내비게이션 메뉴가 웹 사이트에 많이 등장하고 있다. 그러나 이러한 도구를 사용하여 제작한 메뉴나 기존의 웹 내비게이션에 관한 연구에서는 실제로 메뉴 제작에 필요한 기술적인 기법이나 새로운 방법으로서의 접근이 아니어서 대부분의 웹 내비게이션 메뉴가 일률적이고 단조로운 구조를 하고 있다. 이에 본 논문에서는 프로그래밍 기법에 주안점을 두어 보다 기술적으로 접근하는 것을 목적으로 일반적으로 가장 많이 사용되는 풀다운메뉴를 설계하는 새로운 기법을 제안한다. 풀다운메뉴의 특징은 분류해야 할 정보를 메인메뉴에서 먼저 대분류로 나누고, 서브메뉴에서 상세정보를 선택할 수 있는 구조로서, 좁은 공간에서 많은 정보를 표현할 수 있기 때문에 일

* 정회원, 한중대학교 모바일IT공학과

접수일자 : 2016년 01월 26일
수정일자 : 2016년 03월 28일
게재확정일 : 2016년 03월 28일

교신저자 : 조한수, e-mail : hscho@hanzhong.ac.kr

반적으로 많이 사용된다. 풀다운메뉴의 기본적인 동작은 메인메뉴가 선택되면 메뉴의 색상이 변경되거나 혹은 이미지가 롤오버 되어 선택된 메뉴가 강조되고 곧이어 서브메뉴가 펼쳐져서 정보선택을 할 수 있다. 서브메뉴는 몇 단계의 하위 서브메뉴를 포함할 수도 있다. 본 논문에서는 풀다운메뉴의 이러한 기본적인 동작 이외에 새로운 커스터마이즈한 복합적인 기능을 가진 풀다운메뉴 설계 방법을 제안한다. 먼저 메인메뉴의 작은 영역에서도 다양한 기능을 할 수 있도록 메인메뉴를 구성하는 엘리먼트들을 쌓임 규칙을 이용하여 서로 다른 레이어에 배치하고 이 엘리먼트들을 각각 독립적으로 고유의 기능을 수행할 수 있도록 한다. 이를 위해 본 논문에서는 한 개의 메인메뉴를 구성하기 위하여 8개의 엘리먼트를 사용하고, 이들 엘리먼트들이 독립적으로 다른 기능을 할 수 있도록 설계하였다. 또한 서브메뉴 항목을 선택하면 선택 바가 애니메이션 되면서 나타나고 동시에 서브메뉴 항목타이틀도 확대됨으로써 시각적인 측면을 향상하여 정보선택의 효율성을 높였다. 또한, 풀다운메뉴에서 제공하는 정보가 많은 경우 이미 알고 있는 항목 명을 입력함으로써 그 항목이 위치해 있는 메뉴 경로를 자동으로 탐색하는 과정을 보여줌으로써 여러 경로를 재차 탐색해야 하는 불편을 덜어주었다.

이러한 복합적인 기능을 하기 위해서 메뉴를 구성하는 엘리먼트들을 직접 제어하여 동작을 수행할 수 있도록 구현해야 한다. 제안한 방법은 풀다운메뉴이외에 다른 웹 콘텐츠제작에도 쉽게 응용될 수 있기 때문에 이를 이용하면 보다 인터랙티브한 웹 어플리케이션 제작이 가능하다.

Ⅱ. 쌓임 규칙과 애니메이션기법을 이용한 복합기능을 가진 풀다운메뉴

본 논문에서는 쌓임 규칙과 애니메이션 기법을 이용한 복합기능을 가진 풀다운메뉴 설계 방법을 제안한다. 먼저 메인메뉴에 다양한 기능을 추가하기 위해서는 다수의 엘리먼트로 메인메뉴를 구성하고 이들을 서로 다른 레이어에 배치해야 한다. 이렇게 배치된 엘리먼트들은 각각 고유의 기능을 독립적으로 수행할 수 있다.

1. 쌓임 규칙을 이용한 다중 레이어로 구성된 메인메뉴 설계

가. CSS와 jQuery

최근에 인터랙티브한 웹 페이지 제작을 위해 jQuery[9]와 CSS(Cascading Style Sheets)[10]가 많이 사용된다. jQuery는 자바스크립트 라이브러리로 자바스크립트보다 간결하게 인터랙션 및 이벤트처리를 구현할 수 있다. 또한 CSS는 HTML로 작성되어 있는 웹 문서에 스타일을 적용함으로써 문서의 구조와 디자인을 분리할 수 있어서 다양한 레이아웃을 쉽게 제작할 수 있다. 이러한 특징으로 인해 이 두 요소를 함께 사용하면 다양한 인터랙티브한 웹 콘텐츠를 제작할 수 있기 때문에 세계적으로 가장 많이 사용된다. 본 논문에서 제안한 풀다운메뉴를 구현하기 위해 이 두 요소를 사용한다. 먼저 HTML로 풀다운메뉴의 구조를 표현하고 여기에 CSS를 이용한 스타일을 적용하여 수평메인메뉴와 수직서브메뉴의 레이아웃을 구성한다. 또한 메인메뉴를 구성하는 다수의 엘리먼트들에 대하여 쌓임을 제어하여 서로 다른 레이어에 배치하여 이들 각각이 독립적인 고유의 기능을 할 수 있도록 구성한다. 이렇게 구성된 엘리먼트들에 jQuery를 이용한 이벤트처리를 함으로써 제안한 풀다운메뉴가 동작하는 구조로 되어 있다.

나. 쌓임 규칙

HTML로 작성한 웹문서의 각 엘리먼트들은 기본적으로 2차원 평면 위에 배치된다. CSS의 z-index 속성을 사용하면 엘리먼트가 놓이는 2차원 평면에서 z축이 추가되어 여기에 엘리먼트를 배치함으로써 쌓이는 순서를 결정할 수 있다. 이렇게 배치된 엘리먼트는 하나의 쌓임 맥락(Stacking Context)을 형성한다. z-index 속성으로 쌓임 순서를 결정하는 경우에는 반드시 position 속성이 설정되어 있어야 하고 z-index 속성 값이 클수록 위에 놓이게 된다.

메인메뉴에 다양한 기능을 추가하기 위해서는 먼저 다수의 엘리먼트를 사용하여 메인메뉴를 구성해야 한다. 또한, 구성된 엘리먼트 각각이 독립적으로 고유의 기능을 하기 위해서는 서로 다른 레이어에 배치해야 한다. 이러한 배치를 하기 위해 본 논문에서는 다음과 같은 쌓임 규칙[11]을 이용한다.

1. z-index 속성이 설정되지 않은 경우의 쌓임

① position 속성이 지정된 엘리먼트들은 HTML에서 등장하는 순서대로 쌓인다.

② position 속성이 지정되지 않은 엘리먼트는 position 속성이 지정된 엘리먼트보다 아래에 놓인다. 본 논문에서는 쌓임을 형성하기 위해 사용되는 position 속성은 relative와 absolute 방식을 사용한다. relative인 엘리먼트는 현재 엘리먼트가 놓여있는 위치를 기준으로 지정한 오프셋만큼 떨어져있는 곳에 배치되는 방식이고, absolute방식은 position이 지정된 가장 가까운 부모엘리먼트를 기준으로 지정한 오프셋만큼 위치해서 배치되는 방식이다.

2. z-index 속성이 설정되어 있는 경우의 쌓임

① position 속성과 z-index 속성이 지정되어 있는 엘리먼트는 하나의 쌓임 맥락(Stacking Context)을 형성하고,

z-index 속성 값으로 엘리먼트의 쌓임 순서를 제어할 수 있다(z-index 속성 값이 큰 엘리먼트가 위에 쌓인다).

② 하나의 쌓임 맥락은 다른 쌓임 맥락을 포함할 수 있다. 즉, 쌓임 맥락들이 모여서 계층구조를 형성한다.

③ 자식 엘리먼트들의 z-index 속성 값은 오로지 부모 엘리먼트 안에서만 쌓임 경쟁을 할 수 있다. 즉, 부모엘리먼트와 자식엘리먼트 각각에 position 속성과 z-index 속성이 설정되어 있는 경우, 자식 엘리먼트들은 부모엘리먼트를 기준으로 z-index 속성 값에 따라 쌓임 순서가 결정된다.

다. 메인메뉴 구성방법 및 이벤트처리

기존의 풀다운메뉴의 메인메뉴에는 메인타이틀만을 사용하여 사이트에서 제공하는 정보를 대분류하고, 세부정보는 서브메뉴에서 선택할 수 있도록 구성한 형태가 대부분이다. 즉, 분류정보를 표현하기 위해 메인메뉴타이틀에만 의존하고 있기 때문에 메인메뉴에서 분류되어야 할 정보가 많은 경우에는 메인메뉴의 카테고리 정보만으로 필요한 정보가 포함되어 있는지의 여부를 판단하기 어렵다. 따라서 필요한 정보를 탐색하기 위해 서브메뉴 항목 전체를 일일이 마우스로 탐색하여 확인해야 하는 불편이 있다. 이에 본 논문에서는 메인메뉴에 타이틀이외에 상세요약정보를 출력하는 기능을 추가하기 위하여 메인메뉴를 다수의 엘리먼트로 구성하고 이들 각각을 서로 다른 레이어에 배치한다. 메인메뉴가 선택되면 이렇게 배치된 엘리먼트를 이용하여 메인메뉴타이틀에 대한 간략설명이나 서브타이틀 및 서브메뉴에서 제공하는 분류정보에 대한 설명 등에 대한 중분류된 정보를 메인메뉴영역에 출력함으로써 서브메뉴를 일일이 탐색하지 않고 보다 효율적인 정보선택을 할 수 있다. 이러한 중분류된 정보는 메인메뉴에서 자세한 설명이 필요하거나, 혹은 서브메뉴를 포함하고 있는 경우에 출력한다. 경우에 따라서는 고유의 독자적인 기능을 구현하기 위해서는 엘리먼트를 추가하고 이를 이용하여 새로운 기능을 구현할 수 있다. 이러한 레이어를 이용한 구현방식은 중분류된 여러 가지 정보를 추가로 보여주는 역할뿐만 아니라 메인메뉴에서 독자적으로 표현하고자 하는 고유의 기능도 레이어를 형성한 엘리먼트를 이용하여 기능을 구현함으로써 기능성과 시각적인 효과를 함께 갖춘 메뉴를 구현할 수 있는 장점이 있다.

그림 1은 본 논문에서 제안한 풀다운메뉴의 HTML 구조로서 <ul class="mmenu">엘리먼트의 자식엘리먼트인 엘리먼트 각각은 메인메뉴 1개의 형태를 구성하는 역할을 한다. 메인메뉴에 다양한 기능을 추가하기 위하여 문서로딩이 완료되면 HTML DOM을 조작하여 그림 1에서 메인메뉴를 구성하는 각 <li> 엘리먼트에 대하여 그림 2와 같이 엘리먼트를 추가한다. 이 엘리먼트들에 대하여 쌓임 규칙을 적용하여 그림 3과 같은 다수의 레이어를 구성한다. 이렇게 구성된 엘리먼트에 대하여 이벤트처리를 하면 아주 짧은 시간 간격을 두고 각 엘리먼트 고유의 독립적인 기능이 가능해 진다.

```
<ul class="mmenu">
  <li><a href="#">Home</a></li>
  <li><a href="#">Tutorials</a>
    <ul class="submenu">
          :
      <li><a href="#">jQuery Basic</a>
        <ul class="submenu">
          <li><a href="#">Selector</a></li>
          <li><a href="#">Traversing</a></li>
                :
          <li><a href="#">CSS Styling</a></li>
        </ul>
      </li>
      <li><a href="#">Javascript</a></li>
                :
    </ul>
  </li>
          :
</ul>
```

그림 1. 메인메뉴를 구성하는 HTML

```
<li>
  <a href="#">Tutorials</a>
  <span class="title">Tutorials</span>
  <div>
    <span class="desc1">Tutorial Description</span>
    <span class="desc2">SubtitleName</span>
    <span class="desc3">Description</span>
    <span class="desc4"><img … /></span>
  </div>
</li>
```

그림 2. 다중 레이어 구성을 위한 엘리먼트 구조

본 논문의 풀다운메뉴의 각 메인메뉴는 8개의 엘리먼트를 사용하여 그림 3과 같이 레이어를 구성한다. 즉, 엘리먼트에 , , <a> 3개의 엘리먼트가 쌓임을 형성하고 있고, , , , 4개의 엘리먼트가 부모 엘리먼트인 엘리먼트를 기준으로 쌓임을 형성하고 있다.

가장 상위에 있는 레이어의 <a>엘리먼트는 메뉴의 기본적인 기능인 하이퍼링크기능을 한다. 두 번째 레이어의 엘리먼트는 메인메뉴에 마우스가 올라가면 메인메뉴타이틀이 애니메이션으로 확대되면서 위쪽으로 슬라이딩되어 한층 선택된 메뉴를 강조하는 역할을 한다. 또한 이러한 슬라이딩으로 세 번째 레이어의 엘리먼트 안에 포함된 요약정보를 출력하는 4개의 엘리먼트가 메인메뉴로 애니메이션 되어 들어오는 공간을 확보하는 역할을 한다(그림 4의 ②). 세 번째 레이어에 배치된 엘리먼트의 자식 엘리먼트인 4개의 엘리먼트들은 풀다운메뉴가 랜더링된 초기 시점에는 보이지 않도록 메인메뉴 바깥영역에 위치해 있

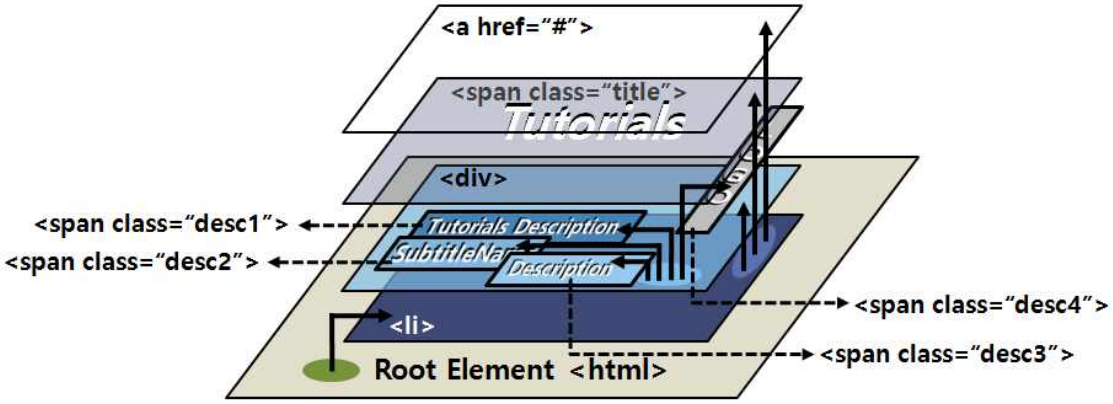

그림 3. 메인메뉴를 구성하는 엘리먼트들의 쌓임 구조

다가 메인메뉴에서 mouseenter 이벤트가 감지되면 짧은 시간 간격 차이를 두고 각각 애니메이션 되어 메인메뉴 안쪽으로 슬라이딩되어 들어온다. 엘리먼트는 각 메인메뉴에 대한 간략한 설명을 보여주는 기능을 한다. 즉 메인메뉴타이틀보다 조금 자세한 설명정보를 사용자에게 보여줌으로써 정보선택 편리성을 제공한다. 이 엘리먼트는 처음에는 좌측 바깥영역에 위치하여 보이지 않다가 메인메뉴에 마우스가 올라가면 메인메뉴 안쪽으로 애니메이션 되어 들어오는 동작을 한다(그림 4의 ③). 엘리먼트는 해당 메인메뉴에서 분류되는 정보의 서브타이틀을 보여주는 역할을 하고 메인메뉴 아래 측 바깥영역에서 위로 애니메이션 되면서 나타난다(그림 4의 ④). 엘리먼트는 서브타이틀에 대한 간략 설명 정보를 포함하고 있다(그림 4의 ⑤, ⑥). 이것은 엘리먼트 우측 옆에서 보이지 않게 위치해 있다가 엘리먼트의 애니메이션이 완료되면 그 위치에서 확대되고(그림 4의 ⑤) 곧이어 축소되는(그림 4의 ⑥) 애니메이션을 함으로써 간략정보를 강조하여 시각적인 측면을 향상시키는 기능을 한다. , 엘리먼트는 서브메뉴를 포함하고 있는 메인메뉴에 마우스가 올라갔을 때에만 동작된다. 곧이어 로고이미지를 역할을 하는 엘리먼트는 메인메뉴 좌측 바깥영역(보이지 않는)에서 안쪽 우측 가장자리까지 애니메이션으로 슬라이딩되면서 나타난다(그림 4의 ⑦). 이러한 일련의 동작을 하는 4개의 엘리먼트들은 서로 다른 레이어에 배치되어 있기 때문에 각각 독립적인 기능이 가능하며, 이러한 동작들은 메인메뉴에 마우스가 올라가면 아주 짧은 시간 간격을 두고 순차적으로 이루어진다. 즉, 이러한 엘리먼트들은 메인메뉴의 역할에 따라 제어될 수 있다. 이러한 동작을 하는 메인메뉴를 구성하는 엘리먼트들은 그림 3과 같이 서로 다른 레이어를 형성하여 각각 고유의 기능을 수행하기 위해서는 쌓임 규칙을 이용한다. 그림 5는 메인메뉴의 형태를 구성하는 엘리먼트 쌓임 맥락에 속하는 3개의 엘리먼트들에 대한 쌓임 순서를 지정하기 위한 스타일이고, 그림 6은 이들 엘리먼트 중에서 요약정보를 애니메이션으로 출력하는 세 번째 레이어에 배치되어 있는 엘리먼트 쌓임 맥락에 속하는 4개의 엘리먼트에 대한 스타일을 지정한 것이다. 먼저, 메인메뉴의 형태를 구성하는 엘리먼트와 그 자식 엘리먼트들에 대하여 각각 position속성과 z-index속성을 지정한다. 이렇게 함으로써 엘리먼트는 하나의 쌓임 맥락을 형성하고 그 자식 엘리먼트들의 쌓임은 부모 엘리먼트 의 쌓임 맥락 안에서 유효하게 되어서 z-index속성 값이 큰 순서대로 즉, <a>, , 엘리먼트 순서로 위에서 아래로 쌓임을 형성한다. position 속성과 z-index 속성이 설정되어 있는 엘리먼트는 하나의 쌓임 맥락을 형성하고, 그 안에 다른

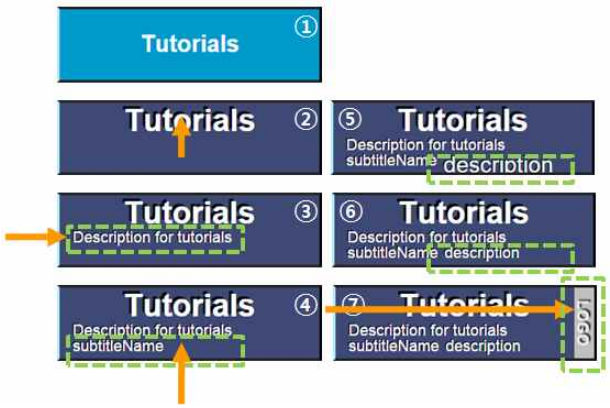

그림 4. 메인메뉴에 mouseenter 이벤트발생 시 메인메뉴를 구성하는 각 엘리먼트의 애니메이션 되는 과정(① 초기 메인메뉴, ② 메인타이틀확대 및 위로 슬라이딩, ③ 좌에서 애니메이션, ④ 아래에서 애니메이션, ⑤ 확대애니메이션, ⑥ 축소애니메이션, ⑦ 좌에서 로고이미지 애니메이션).

```
ul.mmenu > li {
    position: relative; z-index: 1;
}
ul.mmenu > li > a {
    display: block; opacity: 0;
            :
    position: relative; z-index: 3;
}
ul.mmenu > li > span.title {
            :
    position: absolute; z-index: 2;
    left: 50%; top: 50%;
    transform: translate(-50%,-50%);
}
ul.mmenu > li > div {
    position: absolute; z-index: 1;
    top: 0px; overflow: hidden;
}
```

그림 5. 엘리먼트 쌓임 맥락에 속하는 엘리먼트들의 쌓임을 구성하기 위한 CSS

```
ul.mmenu > li > div > span {
    position: absolute;
    text-shadow: -1px -1px black;
}
ul.mmenu > li > div > span.desc1 {
    left: -100%; top: 26px;
    z-index: 1;
}
ul.mmenu > li > div > span.desc2 {
    left: 5%; top: 64px;
    z-index: 2;
}
ul.mmenu > li > div > span.desc3 {
    top: 38px; opacity: 0;
    z-index: 3;
}
ul.mmenu > li > div > span.desc4 {
    left: -15px; top: 0px;
    z-index: 4;
}
```

그림 6. 엘리먼트 쌓임 맥락에 속하는 엘리먼트들의 쌓임을 구성하기 위한 CSS

쌓임 맥락을 포함할 수 있다. 따라서 엘리먼트도 position 속성과 z-index 속성이 설정되어 있기 때문에 하나의 쌓임 맥락을 형성한다. 이 쌓임 맥락 안에서 자식 엘리먼트인 4개의 엘리먼트들을 서로 다른 레이어에 배치하기 위해서 z-index 속성 값을 지정한다. 그러면 엘리먼트 쌓임 맥락 안에서 쌓임 경쟁이 일어나서 z-index 속성 값이 큰 순서로 쌓임을 형성하게 된다. 이때 엘리먼트의 자식 엘리먼트는 z-index 속성 값의 크고 작음에 관계없이 엘리먼트의 쌓임 맥락에 속하는 엘리먼트들과는 쌓임 경쟁을 할 수 없다. 왜냐하면 쌓임 경쟁은 엘리먼트가 속해 있는 쌓임

```
<script>
$(document).ready(function(){
            :
  $("ul.mmenu li").hover(function(e){
            :
    $this.find("span.title")
      .animate({fontSize:"1.5em",top:"25%"},
        function(){/*서브메뉴 처리*/})
      .addClass("highlight_title");

    $this.find("span.desc1")
      .animate({left: "5%"}, function(){
        $this.find("span.desc2")
          .animate({top:"38px"},function(){
            $this.find("span.desc3")
              .css({opacity: 1.0})
              .animate({fontSize: "1.1em"},
                function(){
                  $(this).animate({
                    fontSize: "12px"
                  })
                });
          });
      });

    $this.find("span.desc4")
      .animate({left:$this.width()-15+"px"});
    $this.addClass("active_bgcolor");
  }, function() {
    //mouseleave 처리
  }
}
</script>
```

그림 7. 메인메뉴의 mouseenter 이벤트처리

맥락 안에서만 가능하기 때문이다. 정리하면 쌓임 규칙을 이용하여 2차원 평면상에 놓여 있는 엘리먼트들을 그림3에서와 같이 엘리먼트 쌓임 맥락 안에서 <a>, , 엘리먼트 순으로 쌓임을 형성하고, 엘리먼트 쌓임 맥락 안에서는 , , , 엘리먼트 순으로 쌓임을 형성함으로써 메인메뉴를 구성하는 엘리먼트들은 각기 고유의 기능을 독립적으로 수행하기 위한 레이어에 배치되었다. 그림 3에서의 실선 화살표는 메인메뉴를 구성하는 각 엘리먼트가 어떤 쌓임 맥락에 속해 있는 지를 나타낸다.

지금까지의 내용을 간략히 정리해보면 그림 1에서 풀다운메뉴의 구조를 HTML로 작성하고, 문서가 로딩 되면 그림 2와 같이 DOM을 조작한다. 여기에 그림 5와 6의 CSS를 적용하여 그림 3과 같은 다중 레이어를 구성한다. 즉, 그림 2의 포함관계에 있는 각 엘리먼트들은 평면상에 위치해 있으나 이를 쌓임 규칙을 적용하여 그림 3과 같은 구조의 레이어를 구축하여 각 메인메뉴에 mouseenter 이벤트가 감지되면 메인메뉴타이틀은 위쪽으로 애니메이션으로 확대되면서 요약정보 레이어에 있는 4개의 엘리먼트들은 메인메뉴 안쪽으로 슬라

이딩되어 각각의 기능을 수행하고 동시에 메인메뉴의 형태를 구성하는 엘리먼트의 배경색상을 변경하는 일련의 동작이 순차적으로 아주 빠른 시간간격을 두고 거의 동시에 이루어져서 메인메뉴 선택이 완료된다. 하이퍼링크 레이어가 가장 위에 위치하므로 링크기능도 원활하게 이루어진다. 그림 7은 메인메뉴의 엘리먼트에 이벤트핸들러를 연결하여 mouseenter 이벤트가 발생했을 때 그 처리를 보여주는 jQuery 부분코드로서 .animate() 메서드의 다양한 속성을 이용하여 서로 다른 레이어에 위치해 있는 메인메뉴를 구성하는 각 엘리먼트들을 애니메이션 한다. 이 메서드를 그림 7에서와 같이 각 엘리먼트와 조합하여 사용하면 아주 짧은 시간 간격을 두고 거의 동시에 애니메이션이 가능해 진다. 선택된 메인메뉴에 mouseleave 이벤트가 발생했을 때의 처리는 펼쳐진 서브메뉴가 접히면서 메인메뉴에서 진행되었던 과정이 역순으로 진행된다. 로고이미지는 메인메뉴에 처음 mouseenter 이벤트가 발생할 때에만 메인메뉴 좌측 보이지 않은 영역에서 메인메뉴 우측 가장자리까지 애니메이션 되어 나타나고, mouseleave 이벤트 시에는 우측가장자리 위치에서 우측 바깥 영역으로 들어가서 보이지 않다가 mouseenter 이벤트에 다시 우측 가장자리 위치로 다시 나오는 애니메이션 처리를 하여 너무 현란하게 움직이지 않고 로고이미지의 기능을 간략히 출력할 수 있도록 구현했다. 이와 같이 레이어를 형성한 메인메뉴를 구성하는 엘리먼트들은 아주 짧은 시간 간격을 두고 동작을 제어할 수 있기 때문에 다양한 커스터마이즈한 기능도 메인메뉴에 추가할 수 있다.

2. 애니메이션 기능을 가진 선택 바를 이용한 서브메뉴 설계

가. 서브메뉴 선택 바 구성

메인메뉴에서 mouseenter/mouseleave 이벤트가 감지되면 메인메뉴에 포함된 서브메뉴의 펼침/접힘 처리가 이루어진다. 서브메뉴는 다단계로 구성되며, 서브메뉴 항목을 선택하기 위해 서브메뉴에 마우스가 올라가면 선택 바가 애니메이션 되면서 나타난다. 기존의 대부분의 풀다운메뉴에서의 서브메뉴 선택은 배경색상을 변경하여 선택된 메뉴를 강조하는 것이 일반적이다. 그러나 보다 시각적인 효과를 향상시켜 사용자에게 원하는 정보선택의 효과가 전달되기 위해서는 배경색상 변경에 의한 항목 선택보다 향상된 선택방법이 필요하다. 이에 본 논문에서는 서브메뉴 선택 바를 애니메이션으로 제어함으로써 시각적인 측면을 강조하였다.

서브메뉴의 항목이 선택되면 그림 8에서와 같이 모서리가 둥근형태의 선택 바가 확대 애니메이션 되면서 나타난다.

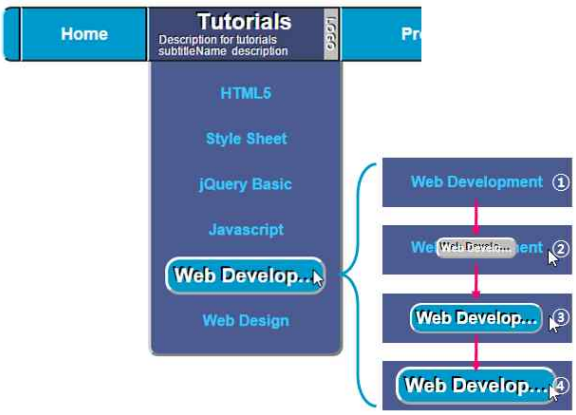

그림 8. 서브메뉴 선택 바 애니메이션

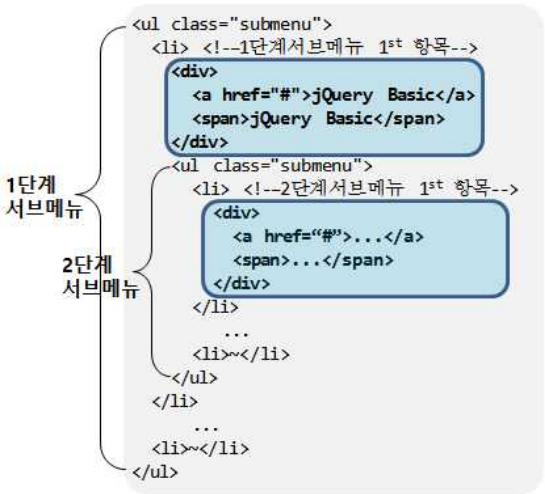

그림 9. 서브메뉴 선택 바 구성

```
/* 서브메뉴선택바 */
ul.submenu li a {
    position: absolute;
    left: 50%; top: 50%;
    transform: translate(-50%, -50%);
    width: 0px;
    height: 0px;
        :
}

/* 서브메뉴 텍스트 */
ul.submenu li span {
    display: inline-block;
    height: 50%;
    width: 80%;
    margin: 12px auto;
        :
}
```

그림 10. 서브메뉴 선택 바와 타이틀을 구성하는 CSS

동시에 서브메뉴 텍스트의 크기도 함께 애니메이션으로 확대된다. 이를 구현하기 위해서는 그림 1의 <ul class="submenu">엘리먼트의 내용을 그림 9에서와 같이 <a>엘리먼트 뒤에 엘리먼트를 추가하고 메뉴텍스트도 복사해 둔다. 그리고 이 두 엘리먼트를 엘리먼트로 감싼다. 서브메뉴에서 <a>엘리먼트는 하이퍼링크기능과 서브메뉴 각 항목에 mouseenter 이벤트가 발생했을 때 항목선택을 비주얼하게 강조하는 선택 바를 구현하기 위해 사용하고, 엘리먼트는 서브메뉴 타이틀을 표시하기 위해 사용된다. 서브메뉴를 선택하기 전에는 <a>엘리먼트가 보이지 않도록 width, height의 크기를 '0px'로 설정해 두고 마우스가 서브메뉴 항목으로 이동되면 그림 8(①~④)과 같이 서브메뉴 각 항목의 정 중앙지점에서 방사형 형태의 애니메이션으로 선택 바가 확대되면서 나타난다. 이것은 서브메뉴 항목의 안쪽 깊은 곳에서 앞쪽 시선방향으로 튀어나오는 효과를 보여준다. 일반적으로 메뉴가 선택되면 색상이 변경되거나 이미지가 롤오버 되는 경우가 대부분이다. 또한 CSS의 속성을 이용한 애니메이션 방법은 속성에서 제공하는 기능이외의 독자적인 고유의 기능으로 확장할 수 없다. 제안한 방법처럼 풀다운메뉴의 서브메뉴 선택 시에 메뉴안쪽에서 보이지 않는 상태에서 애니메이션 되면서 선택 바가 나타나는 방법은 거의 보고된 바가 없다. 이를 구현하기 위해서는 해결해야 할 몇 가지 문제점이 있다. 다음 절에선 이러한 문제점을 분석하여 해결방법을 제시하고 구현 방법에 대해서도 기술한다.

나. 서브메뉴 선택 바 이벤트처리

그림 10에서는 서브메뉴 선택 바 역할을 하는 <a>엘리먼트와 서브메뉴 텍스트를 표현하는 엘리먼트의 스타일을 보여준다. <a>엘리먼트의 position 속성이 absolute이므로 엘리먼트 위에 놓인다. <a>엘리먼트가 확대되어 선택 바가 만들어지기 전까지는 엘리먼트의 메뉴텍스트가 보이게 된다.

서브메뉴 선택 바를 구현하기 위해서 그림 9와 같이 <li> 엘리먼트 안에 포함된 <a>, 엘리먼트를 엘리먼트로 감싸고 이 엘리먼트에 그림 11에서와 같이 mouseenter/mouseleave 이벤트핸들러를 연결시켜 서브메뉴 선택 바를 애니메이션으로 확대하여 나타나게 한다. 엘리먼트로 <a>, 엘리먼트를 감싸서 선택 바 애니메이션 확대 및 축소 처리를 하는 이유는 다음의 두 가지 문제점을 해결하기 위해서이다. 첫 번째는 엘리먼트에 mouseenter/mouseleave 이벤트핸들러를 연결하는 경우 즉, '$("ul.submenu li").hover(handlerIn, handlerOut)' 형식으로 이벤트처리를 하는 경우를 고려해보면 먼저 1단계 서브메뉴

```
/* 서브메뉴선택바 mouseenter, mouseleave */
$("ul.submenu li > div").hover(function(){
   $selectBar = $(this).find("a").first();
   $selectBar.stop(true, false)
      .animate({
         height: "40px",
         width: "170px",
         lineHeight: "34px",
         borderRadius: "15px",
         borderWidth: "3px",
         fontSize: "1.3em"
      }, function(){
         $(this).addClass("active_bg");
      }).addClass("highlight_title");
}, function() {
     /* mouseleave 처리 */
});
```

그림 11. 서브메뉴 선택 바 이벤트 처리

의 항목(<li>)에 마우스를 올리면(mouseenter) 선택 바가 애니메이션 되어 나타나고(ACTIVE) 2단계 서브메뉴를 포함하고 있으면 2단계 서브메뉴가 우측으로 펼쳐진다. 곧이어 마우스를 2단계 서브메뉴 항목으로 이동하면 이동된 항목에 선택 바가 나타나서 ACTIVE 상태로 된다. 이때 일반적인 형태로는 1단계 서브메뉴에서 ACTIVE된 선택 바가 INACTIVE 상태로 되어 사라지고 이동된 2단계 서브메뉴의 항목에만 선택 바가 ACTIVE 상태로 되어야 된다. 그러나 2단계 서브메뉴의 항목으로 마우스가 이동된 영역은 그림 9에서와 같이 1단계 서브메뉴의 엘리먼트의 영역 내부가 되어서 1단계 서브메뉴의 항목과 2단계 서브메뉴의 항목 모두에서 선택 바가 ACTIVE 상태로 남아있게 되는 문제점이 발생한다. 이것은 2단계 서브메뉴의 항목으로 마우스가 이동되더라도 그 영역은 1단계 서브메뉴의 항목 내부영역이므로 1단계 서브메뉴 항목에 마우스가 올라가 있는 상태가 계속 유지되어 1단계 서브메뉴의 항목 선택 바는 ACTIVE 상태로 계속 남아있는 것은 당연한 결과이다. 이러한 이유로 서브메뉴의 엘리먼트에 이벤트핸들러를 연결하여 서브메뉴 선택 바 처리를 할 수 없다. 두 번째 발생하는 문제점으로는 '$("ul.submenu li span").hover(handlerIn, handlerOut)' 형식과 같이 엘리먼트에서 이벤트핸들러를 연결하여 이벤트를 처리한다고 가정해 보면 먼저, 엘리먼트에 mouseenter 이벤트가 발생하면 선택 바 역할을 하는 <a>엘리먼트는 처음에는 보이지 않다가 점점 애니메이션으로 확대되어 나타나면서 마우스가 올라가 있는 엘리먼트를 가리게 되어 엘리먼트에는 의도하지 않은 mouseleave 이벤트가 발생하게 된다. 그 결과로 선택 바는 축소 애니메이션 되어 보이지 않은 원래 상태로 되돌아가는 현상이 발생한다. 즉, <a>엘리먼트는 보이지 않는 상태(width:0, height:0)로 엘리먼트 위에 겹쳐져 놓여있고 서브메뉴

항목이 선택되면 <a>엘리먼트가 확대되어 나타나면서 엘리먼트를 가리게 되어 엘리먼트에 mouseleave 이벤트가 발생하면서 나타나는 문제점이다(만약, <a>엘리먼트에서 이벤트처리를 한다고 가정해보면 <a>엘리먼트는 보이지 않는 상태이므로 이벤트감지 자체가 되지 않는다).

이러한 문제점을 해결하기 위해 본 논문에서는 엘리먼트가 중첩된 경우 바깥쪽(outer) 엘리먼트에서 mouseenter 이벤트가 발생하고 이어서 안쪽(inner) 엘리먼트에 mouseenter/mouseleave 이벤트가 발생하는 경우에 안쪽 엘리먼트에서 발생한 이벤트는 부모 엘리먼트 즉 바깥쪽 엘리먼트로 전파되지 않는 성질에 착안하여 <a>, 엘리먼트를 포함하는 엘리먼트를 구성하고 이 엘리먼트에 mouseenter/mouseleave 이벤트핸들러를 연결시킨다. 이때 그림 10의 스타일에서와 같이 엘리먼트가 엘리먼트에 완전히 포함되도록 엘리먼트의 width와 height 보다 영역을 작게 설정하여 바깥 엘리먼트인 엘리먼트에서 먼저 mouseenter 이벤트가 발생하도록 유도함으로써 이어서 안쪽 엘리먼트에서 발생하는 이벤트는 부모 엘리먼트인 엘리먼트로 이벤트가 전파되지 않도록 한다. 즉, 엘리먼트에서 발생하는 의도하지 않은 mouseleave이벤트가 선택바 애니메이션에는 아무런 영향을 미치지 못하도록 이벤트처리 구조를 변경한다. 이렇게 구조를 변경하면 엘리먼트에서 발생한 mouseenter/mouseleave 이벤트에 대해서만 집중하여 선택 바 애니메이션을 처리할 수 있다. 또한 서브메뉴의 하위 서브메뉴로 마우스가 이동될 경우에도 엘리먼트에서 mouseleave 이벤트가 발생하므로 ACTIVE된 선택 바는 쉽게 INACTIVE 상태로 변경할 수 있다. 그리고 마우스가 올라간 하위 서브메뉴의 엘리먼트에서는 mouseenter 이벤트에 대한 ACTIVE 처리를 하면 선택 바 기능은 정상적으로 잘 동작하게 된다. 이와 같은 선택 바 처리는 보이지 않게 설정한 엘리먼트가 애니메이션으로 나타나면서 발생하는 이벤트에 대한 처리기법으로 서브메뉴 선택 바 구현에서 뿐만 아니라 다른 기능에서도 적용될 수 있다.

다. 서브메뉴 항목 검색 기능

풀다운메뉴에서 대부분의 정보선택은 서브메뉴 항목에서 이루어진다. 분류해야 할 정보가 많은 경우에 이미 검색했던 정보에 대한 카테고리나 항목의 위치를 정확히 알지 못해서 일일이 재차 검색해서 선택해야 하는 경우가 빈번히 발생한다. 본 논문에서는 서브메뉴 항목을 효율적으로 검색하기 위한 편리한 하나의 수단으로 검색필드를 통하여 서브메뉴타이틀을 입력하면 마우스로 서브메뉴를 검색하는 것과 동일하게 메인메뉴에서 시작하여 검색항목이 속해 있는 하위 서브메뉴까지의 펼침을 자동으로 보여주는 기능을 제공함으로써 사용자가 일일이 탐색하여 선택하는 것보다 효율적으로 서브메뉴를 선택할 수 있다. 이러한 기능을 구현하기 위해 메인메뉴 바 윗부분에 검색필드와 버튼을 만들고, 서브메뉴 타이틀이 입력되면 HTML DOM 상에서 모든 엘리먼트의 하위 엘리먼트인 엘리먼트의 텍스트와 패턴매칭을 수행한다. 매칭이 이루어지면 해당 엘리먼트의 부모엘리먼트를 찾는 일을 루트엘리먼트에 도달할 때까지 반복하여 검색하려는 서브메뉴 항목이 속해있는 부모엘리먼트에서부터 조상엘리먼트들을 임시로 저장해 둔다. 이러한 작업이 완료되면 저장된 엘리먼트들에 대하여 .triggerHandler("mouseenter") 메서드를 이용하여 mouseenter 이벤트를 자동 발생시켜 마우스를 올렸을 때와 동일하게 메인메뉴에서 시작하여 검색하려는 서브메뉴타이틀이 속해있는 메뉴까지 차례대로 펼침을 수행함으로써 사용자가 검색하려는 서브메뉴 항목이 어떤 부모엘리먼트에 속해 있는지를 미리 알 수 있도록 한다.

III. 구현 및 고찰

본 논문에서 제안한 풀다운메뉴 설계 방법은 기존의 미리 준비된 이미지를 사용한 롤오버기능이나 색상변경과 같은 단조로운 구현방법에 비하여 앞장에서 설명한 기술적인 접근 기법에 주안점을 두어 구현하였다. 먼저, 타이틀만을 사용하여 카테고리를 분류하는 기존 메인메뉴 구현방법에 비해, 본 논문에서 제안한 방법은 메인메뉴의 한정된 영역에서 다양한 기능을 할 수 있도록 하기 위해 쌓임 규칙을 적용하여 메인메뉴를 구성하는 각 엘리먼트를 서로 다른 레이어에 배치하여 이들 엘리먼트가 독립적으로 고유의 기능을 할 수 있도록 계층적 레이어를 구성하는 기법을 확립하고, 이를 이용하여 메인메뉴의 다양한 상세 요약정보를 애니메이션으로 출력하도록 메인메뉴를 구현하였다. 이러한 기법은 메인메뉴를 선택할 때 각 레이어에 배치된 엘리먼트가 애니메이션 되는 시각적인 효과 뿐만 아니라 서브메뉴에서 제공하는 상세정보를 미리 메인메뉴 영역에 출력함으로써 하위 단계의 서브메뉴를 일일이 탐색하지 않고 보다 효율적인 정보선택을 할 수 있다.

이러한 기능을 구현하기 위해 본 논문에서는 메인메뉴에 대해 쌓임 규칙을 적용하여 하이퍼링크기능을 하는 레이어, 메인메뉴 타이틀 확대 레이어, 요약정보들을 애니메이션으로 출력하는 레이어, 그리고 메인메뉴 형태를 구성하는 엘리먼트 레이어를 계층적으로 구성하였다. 또한, 요약정보 출력 레이어 내부에는 4개의 엘리먼트가 또 다른 레이어를 형성하도록 구성하고, 메인메뉴가 선택되었을 때 이러한 엘리먼트들이 각각 고유의 기능을 독립적으로 수행할 수 있도록 구현하였다. 본 논문에

그림 12. 복합기능을 가진 풀다운메뉴

서는 상세요약 정보를 애니메이션으로 출력하는 기능 이외에 로고 이미지도 함께 애니메이션으로 출력하여 버튼 이벤트처리 기능을 추가할 수 있도록 구현하였다. 이와 같이 구성한 레이어 구조에 엘리먼트를 추가하는 것만으로도 새로운 독자적인 기능을 쉽게 부착할 수 있기 때문에 기존의 풀다운메뉴에 비해 확장성인 측면에서도 차별성을 부여하였다. 본 논문에서는 메인메뉴에 대한 요약 정보를 애니메이션으로 출력하는 용도로 사용했지만 경우에 따라서는 메인메뉴 선택에 필요한 커스터마이즈한 기능을 추가할 수 있다.

또한, 본 논문에서는 기존의 배경색상 변경으로만 서브메뉴를 선택하는 단순한 처리 방법에 비해 보다 시각적 효과를 강조하기 위해 선택 바 애니메이션 기법을 제안하여 이를 이용하여 서브메뉴를 선택할 수 있도록 구현하였다. 이 기법은 처음에는 선택 바가 보이지 않는 상태에서 서브메뉴 항목에 마우스가 올라가면 선택된 서브메뉴의 중앙 안쪽에서 시선방향으로 선택 바가 확대 애니메이션이 진행되면서 서브메뉴를 강조하는 효과를 나타낸다. 이러한 기법은 선택 바 역할을 하는 엘리먼트와 서브메뉴 텍스트를 표현하는 엘리먼트 간 이벤트처리에서 발생하는 문제점을 해결함으로써 가능하다. 즉, 선택 바가 애니메이션 될 때 발생하는 의도하지 않은 mouseleave 이벤트를 제거하기 위하여 이벤트처리 구조를 변경하는 기법을 사용하여 선택 바 애니메이션 기능을 구현함으로써 기존의 단순한 서브메뉴 처리와 차별성을 두었다. 이러한 선택 바 애니메이션은 그림 11과 그림 12에서와 같이 선택 바의 너비와 높이, 수직정렬위치, 선택 바 테두리의 둥근 정도와 두께, 폰트 크기에 대한 애니메이션이 동시에 진행되고, 이러한 애니메이션이 완료되는 시점에서 선택 바 배경색상 변경과 함께 서브메뉴 항목 타이틀 확대 애니메이션도 아주 짧은 시간 간격을 두고 거의 동시에 진행되어 서브메뉴 선택 바 애니메이션 처리는 완료된다.

또한, 풀다운메뉴에서 분류해야 하는 서브메뉴 항목이 많은 경우, 한 번 검색한 항목이라고 할지라도 그 항목이 어떤 카테고리의 몇 단계 서브메뉴에 있는지를 쉽게 파악하기 어려워 서브메뉴 각 항목을 일일이 재차 찾아들어가는 단순한 처리를 반복해야 한다. 이에 비해 본 논문에서 구현한 자동 검색 기능은 서브메뉴타이틀을 어느 정도라도 인지하고 있으면 검색시간을 현저히 줄일 수 있는 유용한 기능이다. 이 기능은 검색필드에 검색하고자하는 항목타이틀을 입력하면 마우스로 검색하는 것과 동일하게 입력된 항목이 속해있는 상위 서브메뉴를 자동적으로 펼쳐서 검색하고자 하는 항목이 어디에 위치해 있는가의 정보를 사용자에게 미리 제공함으로써 모든 카테고리를 일일이 재차 검색하지 않고 효율적인 재검색이 될 수 있도록 구현하였다.

본 논문에서 제안한 기법의 유효성을 확인하기 위해 HTML을 사용한 경험이 있는 그룹(10명)과 인터넷만 주로 이용하는 그룹(10명)을 대상으로 기존의 단순한 풀다운메뉴와 제안한 기법으로 구현한 풀다운메뉴를 직접 사용하게 하여 기능적인 측면과 시각적인 측면에 관한 설문조사를 실시하였다. 평가결과, 두 가지 측면에 대해 각각 18명, 17명이 제안한 방법이 우수하다는 만족한 결과를 얻었다. 또한 서브메뉴 검색 기능을 설명하지 않은 상태에서 탐색시간을 측정해본 결과, 서브메뉴 항목 당 제안한 방법이 평균 0.3초 빠르게 검색한 좋은 결과를 얻었다. 검색평가를 위해서 2단계에서 3단계 서브메뉴를 구성하였으나 검색항목이 많아지면 많아질수록 더 나은 결과를 얻을 수 있을 것으로 판단된다.

결론적으로 본 논문에서 제안한 풀다운메뉴는 기존의 풀다운메뉴의 기본적인 동작인 대분류기능과 상세분류기능 이외에 쌓임 규칙에 기반한 계층적 레이어를 구성하는 기법을 이용함으로써 메인메뉴에 다양한 기능을 구현했을 뿐만 아니라 새로운 독자적인 기능추가도 풀다운메뉴의 전체적인 구조 변경 없이 레이어를 추가하는 것만으로 쉽게 구현할 수 있는 기

법을 확립하였다. 또한 기존의 단순한 서브메뉴 선택처리 방법에 비해 이벤트처리 구조를 변경하여 보이지 않게 설정한 엘리먼트에 대한 다양한 속성을 사용하여 애니메이션을 진행함으로써 선택한 서브메뉴를 시각적으로 강조하는 기법은 웹 기반 어플리케이션 구현에도 바로 응용될 수 있는 장점이 있다. 아울러 자동 검색 기능도 탑재하여 서브메뉴 깊이가 큰 경우 검색시간을 단축할 수 있도록 검색 효율성을 향상하였다.

본 논문에서 제안한 풀다운메뉴의 스타일은 CSS3로 작성되었고, jQuery는 2.1.1버전으로 테스트하였다. 풀다운메뉴의 전체적인 레이아웃은 그림 12와 같다. 각 메인메뉴의 폭은 카테고리 타이틀문자에 따라 유동적이 될 수 있도록 구현하였고, 또한 메인메뉴 개수나 서브메뉴의 깊이와 같은 풀다운메뉴의 구성요소를 변경하거나 확장하기 위해서는 HTML만 변경함으로써 작동될 수 있도록 프로그래밍 되었기 때문에 개발자가 아니더라도 확장 및 관리가 손쉽게 이루어지도록 구현하였다.

IV. 결 론

본 논문에서 제안한 풀다운메뉴 설계 방법은 기존의 방법에 비해 프로그래밍기법에 기반을 두고 구현하였기 때문에 새로운 기능추가를 위해서는 메뉴 전체구조를 변경하지 않고 해당 엘리먼트를 추가함으로써 그 기능을 확장할 수 있다. 기존의 풀다운메뉴에서 메뉴 선택 시 색상변경 및 롤오버 기능과 같은 단조로운 구현방법에 비해 제안한 방법은 다수의 엘리먼트를 사용하여 메인메뉴를 구성하고 이들 엘리먼트에 쌓임 규칙을 이용하여 서로 다른 레이어에 배치하여 엘리먼트를 직접 제어함으로써 아주 짧은 시간 간격을 두고 엘리먼트 고유의 독립적인 기능을 할 수 있다. 본 논문에서는 각 메인메뉴에 대하여 8개의 엘리먼트에 대하여 레이어를 구성하고 각각 커스터마이즈한 기능을 독립적으로 할 수 있도록 구현하였다. 또한, 풀다운메뉴에서 실제 정보선택이 이루어지는 서브메뉴에 애니메이션 기능을 갖춘 선택 바 기능을 구현함으로써 기존의 서브메뉴 구성방법에 비해 보다 시각적인 측면을 강조할 수 있다. 서브메뉴에서 표현할 정보가 많은 경우를 고려하여 자동 서브메뉴 항목 탐색기능을 탑재하여 사용자가 재탐색 할 경우 편리성을 향상하였다. 본 논문에서 제안한 기법을 이용하여 풀다운메뉴를 구현한 결과 기능적인 측면과 시각적인 측면에서 좋은 결과를 확인하였다. 또한, 커스터마이즈한 기능도 쉽게 추가하여 확장할 수 있기 때문에 그 활용성이 클 것으로 기대되고 아울러 다양한 응용도 가능할 것으로 판단된다.

References

[1] Schwartz, J. P., & Norman, K. L., "The importance of item distinctiveness on performance using a menu selection system", *Behaviour and Information Technology*, 5(2), 173-182, 1986.

[2] Nielsen, J. Designing web usability, Indianapolis, *New Riders Publishing*, 2000.

[3] Park, J., & Kim, J., "Contextual navigation aids for two World Wide Web systems", *International Journal of Human-Computer Interaction*, 12(2), 193-217, 2000.

[4] Yoo, B., & Kim, J., "Experiment on the effectiveness of link structure for convenient cybershopping", *Journal of Organization Computing and Electronic Commerce*", 10(4), 241-256, 2000.

[5] 유병민, “인터넷 정보탐색 과정에서 정보구조와 메뉴디자인의 상호작용 분석”, *정보처리학회논문지*, 제12-B권, 제4호, pp.473-478, 2005년 8월

[6] 배윤선, 이현주, “고령자의 사용편의성을 위한 웹 네비게이션 디자인에 관한 연구”, *디자인학연구*, 통권 제63호, Vol 19, No. 1, pp. 129-140, 2006년

[7] 박은정, 이건표, 서종환, “감성기반 웹 사이트 네비게이션 디자인을 위한 디지털 스토리텔링의 활용”, *디자인학연구*, 통권 제87호, Vol. 23, No. 1, pp. 174-185, 2010년

[8] Snowberry, K., Parkinson, R., & Sisson, N., "Effect of help fields on navigating through hierarchical menu structures", *International Journal of Man-Machine Studies*, 22, 479-497, 1985.

[9] http://jquery.com

[10] http://www.w3.org/Style/CSS

[11] https://developer.mozilla.org/en-US/docs/Web/CSS

저 자 소 개

조 한 수(정회원)

1987년 부산대학교 계산통계학과 학사 졸업.

1990년 성균관대학교 정보처리학과 석사 졸업.

1999년 法政대학 공학연구과 박사 졸업.

1986년 ~ 1993년 한화정보통신 중앙연구소 주임연구원

현재 한중대학교 모바일IT공학과 부교수

<주관심분야 : 패턴인식, 컴퓨터비전, 모바일 및 웹정보처리>