**REPORT**



|  |  |
| --- | --- |
| **과목명** | 웹 프로그래밍 |
| **과제명** | 과제 #1 |
| **학과** | 컴퓨터 공학부 |
| **학번** | 201911247 |
| **이름** | 남권표 |
| **제출일자** | 2020.04.24 |

|  |
| --- |
| 1. **레이아웃 도안** |
| <div id=master-title>  <div id=my-information>  <div id=search-box>  <div id=move-to-article>  <article id=article1>  <aside id=main-aside>  <article id=article2>  <article id=article3>  <article id=article4>  <div id=last-comment>  1번째 큰 상자: <header>  2번째 큰 상자: <nav id=main-navigation>    3번째 큰 상자: <div id=content>  main-section1 -> article1을 담고 있음.  main-section2 -> article2, 3, 4를 담고 있음.  4번째 큰 상자: <footer>  **[결과]** |
| **2. 주요 기능** |
| **[1] Tab-bar 기능.**  **\*Tab-bar 구현 방법**  aside영역안에 radio타입의 input태그 2개를 생성했다. 이후, 두 개의 input태그 각각의 id를 가리키는 label태그를 생성하였고, 첫 번째 div영역은 Go to Articles의 메뉴들을, 두 번째 div영역은 Open Articles의 메뉴들을 담았다. 이때, Go to Articles의 메뉴들은 기사로 바로 접근하게 해주는 기능을 가지고, Open Articles는 클릭 시 해당 주제에 대한 상세 내용을 출력해 주는 기능을 가진다.  label들을 생성했기 때문에 input태그는 보이지 않게 display: none; 을 해준다. input태그가 checked상태가 아니라면 메뉴도 보이지 않아야 하기 때문에 **nth-of-type을 사용**해 해당하는 div영역을 선택하여 **display: none;** 을 사용해주었다. 이후 input태그 가 checked 상태가 되면 메뉴를 보여주기 위해 **display: block;** 을 사용하여 메뉴를 출력하였다. 각 메뉴들의 제목 부분인 label에는 **block속성을 부여**해주어 **width와 height을 지정**해 주었고, **line-height와 text aline: center;** 를 사용하여 각 블록에 중앙에 텍스트를 배치했다. 이후 **float: left;** 덕분에 이들이 수평으로 배치되고, float속성을 사용하였기 때문에 **부모**인 section.tab-title에 **overflow: hidden;** 을 사용해주었다. 또한 두 input태그 중 하나가 checked된 상태라면 **배경색은 #003458, 글씨색은 하얀색**으로 바꾸어 주었다.  이후 aside블록이 넘쳤기 때문에, 스크롤을 사용하기 위해 **overflow-y: scroll;** 을 사용하여 scroll기능을 추가해준다.  **(1) 메뉴의 글블록에 마우스를 올리면 글자 색이 변함.**    왼쪽의 사진은 글자위에 마우스를 올리고 캡쳐 한 사진이다. 마우스를 올린 후 캡처한 사진으로 마우스 포인터는 캡처에 이용했기 때문에 마우스 포인터는 보이지 않는다. 상단에 <li>태그는 Open Articles 메뉴의 리스트들 중 하나이며 왼쪽의 1, 2, 3, 4는 모두 <li>태그 안에 있는 요소들이다. 이때. description은 <strong>(제목)</strong>이 들어있는 클래스로, .description위에 마우스가 올라갔을 때 글자색이 변하도록 하였다. Go to Articles의 메뉴들 블록영역에 마우스가 올라가면 글자색이 변한다.  **(2) 메뉴를 클릭했을 때(사진 or 글), 해당 주제에 대한 상세 내용 출력.**  Open Articles메뉴에서 1번 메뉴를 선택하면 해당 기사인 article1의 내용을 출력한다.  메뉴 2를 눌러도 올바르게 작동하는 것을 확인할 수 있다.  해당 기능은 input태그와 label태그를 이용한 방식으로 코드는 아래와 같다.  모든 article태그안에 ‘radio’타입의 input태그를 생성한다. 이후 Open Articles 메뉴들인 각 li태그에 올바른 input태그의 id를 사용하여 label태그를 사용한다. 위의 사진은 article1의 예시이다. 이후 css를 사용하여 생성한 radio타입의 input을 보이게 하지 않기 위해  **article > input:nth-of-type(1){display: none;}**을 사용한다. 이후 input태그와 동위 레벨에 있는 div중 두 번째에 위치한 div가 article-body이기 때문에, **article > input:nth-of-type(1) ~ div:nth-of-type(2){display: none;}**을 사용하여 article-body의 내용을 가려주었다. 이후 만약 label을 통해 input태그가 체크된 상태가 되면 **article > input:nth-of-type(1):checked ~ div:nth-of-type(2){display: block;}**를 사용하여 해당 기사의 내용을 출력하도록 하였다.  모든 article태그는 article1과 같이 크게 input과 2개의 div로 나누어진다. 그 결과 아까의 css를 통해 모든 기사에 이와 같은 디자인을 적용할 수 있게 된다.  따라서 이전 레이아웃 도안 **[결과]**에서 아무런 기사의 내용이 출력되지 않은 이유는 이 코드들 때문이라고 할 수 있고, 각 메뉴들이 올바르게 동작하는 것을 확인할 수 있다.  **(3) 메뉴를 클릭했을 때(사진 or 글), 각각의 주제에 바로 접근.**  해당 div는 Go to Articles의 메뉴들을 모아 놓은 곳이다. 이때 li태그 안에는 a태그가 존재하는데, 이는 각 메뉴에 맞는 기사의 id를 가리키게 하였다. 해당 li는 article1을 가리키는 기사로 이를 클릭하게 되면 article1로 이동하게 된다.    **[2] nav의 #move-to-article**  nav블록에도 tab-bar의 Go to Article처럼 각 article에 바로 접근할 수 있는 기능이 있다.  해당 부분이 바로 nav블록이다. 이때 article1을 클릭하면 article1으로 이동하고, article3를 이동한다면 article3의 위치로 이동한다. article3로 이동한다면 다음과 같은 화면이 된다.  nav는 이와 같이 2개의 div로 이루어지고 우리가 살펴볼 기능은 move-to-article의 기능이다. move-to-article은 4개의 기사 각각에 이동 가능한 a태그가 존재한다. 이 태그들은 위와 같이 block특성을 부여하여 height와 width를 지정해 주었고, **line-height**와 **text-align: center**를 이용하여 텍스트를 중앙에 배치시켰다. **float: left**를 통하여 위에서 본 nav와 같이 서로 붙어서 출력되며 a블록 구역에 마우스가 올라가면 글씨의 색이 변하게 된다.  **[3] 각 제목을 클릭 시 해당 기사의 내용 출력**  위에서 nav를 통해서 article3로 이동한 경우를 살펴보았다. 이때 기사가 출력이 되지 않았고, 옆에 tab-bar도 없어 기사를 출력하려면 다시 tab-bar로 올라가야 한다. 이런 번거로움을 방지하기 위해 제목을 클릭해도 해당 기사의 내용이 출력하도록 하였다. 이전의 상황에서 article3의 제목을 클릭하면 다음과 같이 기사가 출력된다.  위와 같이 이전에 설명한 tab-bar의 Open Articles와 동일한 css를 사용하여 기사 제목에 해당하는 article-header에 label태그를 사용하면 된다. 또한 .article-header > label > h1:hover {color: #00806a;}을 사용하여 제목위에 마우스가 올라가면 글자색이 바뀌도록 설정하였다.  **[4] 글씨체**  웹 폰트는 위와 같이 추가해주었다.  첫 번째 글씨체는 메인 제목과 footer의 글씨체로 사용하였다.  두 번째 글씨체는 기사 제목의 글씨체로 사용하였다.  세 번째 글씨체는 article2 기사의 글씨체로 사용하였다. article2기사 형식이 채팅이기 때문에 더 귀여운 느낌의 폰트를 사용하였다.  [5] input과 text가 focus됐을 때 보이는 테두리 제거  input의 타입이 text이고 focus 되었다면 배경색이 바뀌고, 원래 보이던 테두리가 사라지게 된다. 만약 outline:0; 을 사용하지 않는다면 밑과 같은 결과가 나온다. submit역시 focus될 때 이러한 테두리가 발생하지 않도록 설정하였다.  => 텍스트 주변 파란색 테두리가 생김.  **[6] (추가 사항) nav가 움직이는 기능을 추가**  처음 웹 페이지를 만들었을 때는 nav가 움직이지 않아 상단에서 멀어지게 되면 tab-bar와 nav를 이용하기에 어려움이 있었다. 즉, tab-bar와 nav는 상당에 있기 때문에 상단에서 먼 article3로 간 경우 다른 기사로 이동할 때 직접 스크롤을 이용해야 한다는 번거로움이 있었다. 따라서 아래와 같이 코드를 변경하여 nav와 header가 움직일 수 있도록 변경하였다.  하지만 이 코드만 추가하는 경우, 한 가지 문제점이 생긴다. 만약 nav에서 article2를 클릭하면, article2의 자리로 이동하지만 브라우저에서 고정위치인 header와 nav때문에 article의 일부가 가려진다. 이러한 문제를 해결하기 위해 각 article의 위치에서 세로로 141px만큼 떨어진 지점에 새로운 id속성을 지정하려는 계획을 세웠다.  해당 코드가 바로 위에서 말한 계획을 코딩으로 표현한 것이다. 각각의 article 바로 위에 for-fake라는 클래스를 가진 div를 만들어 준다. 단, 이 div는 칸을 차지하지 않게 height=0px로 설정해 준다. 그다음 해당 div의 자식으로 article-fake라는 클래스와 고유한 id를 가진 div를 만들어 준다. 이 div를 for-fake로부터 위로 141px만큼 떨어진 위치에 자리하게 된다면, 해당 div는 article에서 위로 141px만큼 떨어진 지점에 위치하게 된다. (141px만큼 떨어뜨리는 이유는 header와 nav의 높이의 합이 141px이기 때문이다.) 이를 위하여 부모 div에는 relative 속성을 부여해 주고, 자식 div에는 absolute 속성을 부여하여 밑에서부터 올라가도록 css에서 설정하였다. 이렇게 article-fake 클래스의 div들을 만들어 주고 아래와 같이 전에 #article(숫자)의 형식의 값을 넣었던 href 속성들에 #article(숫자)-fake 형식의 값으로 바꿔주면 문제를 해결할 수 있다. |
| **3. 토의 사항** |
| [1] 알게 된 점  **(1) 위치 이동시 바뀌는 주소.**  만약 article1으로 이동하기 위해 tab-bar나 nav의 메뉴를 클릭하면 원래 주소에 #article1이 붙는다는 사실을 알게 되었다.  **(2) submit타입의 input에 배경 설정하기.**  위와 같이 submit타입 input의 value 속성에는 아무것도 넣어주지 않는다. 이후 css에background-image를 사용하여 이미지를 넣어주고 background-repeat을 사용하여 반복을 막는다. 이 상태에서는 블록 정중앙에 배치가 안되므로 background-position: center center를 하면 정중앙에 이미지를 넣어준다.    **(3) 1줄 이상의 글을 박스안에 line-height으로 수직 정렬하면 올바른 결과를 출력하지 못한다.**  만약 왼쪽과 같은 결과를 출력하려고 수직 정렬을 line-height을 사용한다면 전혀 다른 결과인 오른쪽 결과가 출력될 것이다. 따라서 왼쪽과 같이 중앙에 정렬하기 위해 margin과 padding을 적절하게 사용하여 수직 정렬을 해주었다.  **[2] 느낀 점**  수업시간에 교수님께서 정해주신 틀대로만 코딩을 하다가 틀, 디자인, 구성 등 모든 것을 내가 정하고 그에 맞게 코딩하는 것이 매우 힘들었다. 하지만 내가 생각한 디자인대로 코딩을 해보면서 잘못 생각하고 있던 부분들을 발견하고, 새로운 사실들을 알게 되는 과정들이 매우 의미 있었다고 생각한다. 지금까지 강의시간에 배웠던 내용들을 활용하면서 복습하는 것 역시 내 부족한 실력을 보완하는데 도움이 됐다. 처음 html을 접하고 나의 첫 웹 프로그래밍 작품을 완성하기까지 투자한 시간이 매우 의미 있었음을 느꼈다. 이후에 javascript를 배우고 css와 html과 함께 사용하며 더 멋진 작품을 만들어보고 싶다. |