CSI2109-01
2018-01

인터넷 프로그래밍
Project 2

보고서

차례

2012147562
최인호

테스트 브라우저 : Chrome 최신버전 (버전 66.0.3359.139(공식 빌드) (64비트))

1. 입력데이터 유효성 검사(Javascript 코드에 대한 설명)

과제 요구사항 : JavaScript를 이용하여 회원 가입시 입력한 form 데이터의 유효성을 클라이언트 사이드에서 검증한다.

회원가입을 위해 사용자로부터 입력받는 데이터는 총 5가지이다. 과제에서 기본으로 요구하는 데이터인 사용자ID, 비밀번호, 생년월일, 휴대전화번호와, 추가적으로 E-mail 주소를 입력받고 유효성 검사를 적용한다. 각 데이터별 유효성검사 규칙과, 검사를 구현한 JavaScript 코드를 이어서 설명한다.

- **사용자 ID**

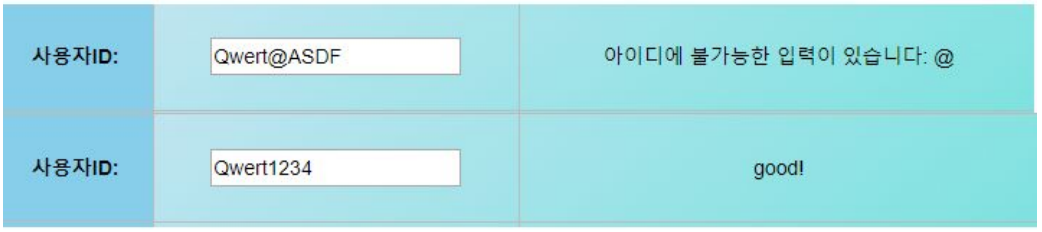
사용자 ID의 유효성 검사 조건은 다음과 같다.
“ID는 6~10자의 영문 대소문자, 숫자와 특수기호(_)만 사용 가능하고, 적어도 1개의 숫자와 대문자를 포함한다.”
이 유효성 검사를 위해 작성한 JavaScript 코드는 다음과 같다.

```
128             <script>
129               var input1 = document.getElementById('userid'); //userid element를 HTML DOM API를 사용하여 input1 변수에 assign
130               //input1.addEventListener('change', function(){document.getElementById('idval').innerHTML="good!"; alert('hi1'); });
131               input1.addEventListener('change', function(){ validation1(event.target.value); });
132               function validation1(value){//userid element의 값이 change됐을때 event처리
133
134                 var check1=false, check2=false; // check1은 숫자가 적어도 한 개 이상 포함되었는지 체크하기 위한 boolean변수
135                 // check2는 대문자가 적어도 한 개 이상 포함되었는지 체크하기 위한 boolean변수
136
137                 //userid의 길이(6~10자) 검사.
138                 //document.getElementById('idval').innerHTML=event.target.value.length;
139                 if (value.length<6 || value.length>10) {
140                     document.getElementById('idval').innerHTML="사용자ID 길이는 6자~10자만 가능합니다. 현재길이 : " + value.length;
141
142                     return false;
143                 }
144                 for (i = 0; i < value.length; i++) { //영문 대소문자, 숫자와 특수기호(_)만 사용 가능 조건 검사
145                   ch = value.charAt(i)
146                   if (!(ch=='_') && !(ch >= '0' && ch <= '9') && !(ch >= 'a' && ch <= 'z') && !(ch >= 'A' && ch <= 'Z')) {
147                     //사용자 ID 데이터로 입력 가능한 모든 문자들을 제외한 다른 문자가 입력됨.
148                     document.getElementById('idval').innerHTML="아이디에 불가능한 입력이 있습니다: " + ch;
149                     return false;
150                   }
151                   if(ch >= '0' && ch <= '9'){ // 적어도 한 개의 숫자 포함
152                     check1 = true;
153                   }
154                   if(ch >= 'A' && ch <= 'Z'){ // 적어도 한 개의 대문자 포함
155                     check2 = true;
156                   }
157                 }
158                 if(check1 == false){ // 데이터 중 숫자데이터가 한 개도 존재하지 않음.
159                   document.getElementById('idval').innerHTML="적어도 한 개의 숫자를 포함하세요";
160                   return false;
161                 }
162                 if(check2 == false){ // 데이터 중 대문자 데이터가 한 개도 존재하지 않음.
163                   document.getElementById('idval').innerHTML="적어도 한 개의 대문자 포함하세요";
164                   return false;
165                 }
166                 document.getElementById('idval').innerHTML="good!"; // 모든 유효성 검사 조건을 통과하면 good! 메세지를 출력함
167                 return true;
168               }
```

[사용자 ID 유효성검사 JavaScript 코드] - register.html, line 128 ~ 168

validation1 함수는 회원가입 폼에서 사용자ID 입력란에 입력된 데이터를 parameter로 전달받는다. 우선 가장 먼저 사용된 if문의 조건은 “if (value.length<6 || value.length>10)”로서, 입력된 데이터의 길이가 6보다 작거나 10보다 큰지 검사한다. 유효성 조건에서 사용자ID는 6~10자이므로 이 범위를

벗어날 경우 현재 입력된 길이와 함께 사용자에게 조건을 명시한다. 다음은 for문을 이용하여 입력값의 처음부터 끝까지 불가능한 입력이 있는지 조사한다. 사용자ID는 영문 대소문자, 숫자, 특수문자(_)만 사용 가능하므로 이 외의 입력값이 있을경우 사용자에게 그 값을 알린다. “if (!(ch=='_') && !(ch >= '0' && ch <= '9') && !(ch >= 'a' && ch <= 'z') && !(ch >= 'A' && ch <= 'Z'))” 조건을 사용하면, 가능한 모든 입력값들을 제외한 다른 값이 들어온 경우를 찾아낼 수 있다. 다음으로 적어도 1개의 숫자와 대문자를 포함하는 조건이다. for문을 사용하여 입력값의 처음부터 끝까지를 조사하면서 값이 숫자인 경우와 대문자인 경우에 각각 check1, check2의 값을 true로 assign해준다. 그리고 유효성검사 끝부분에 check1과 check2가 모두 true인 경우에만 두가지 값이 모두 적어도 한 번은 입력됐다는 뜻이므로 유효성검사를 통과시킨다. 각 조건에 대해 만족하지 못하는 경우에 false를 return하여 유효성 검사를 만족하지 못함을 알린다.



[사용자 ID 유효성검사 웹페이지 동작]

- **비밀번호**

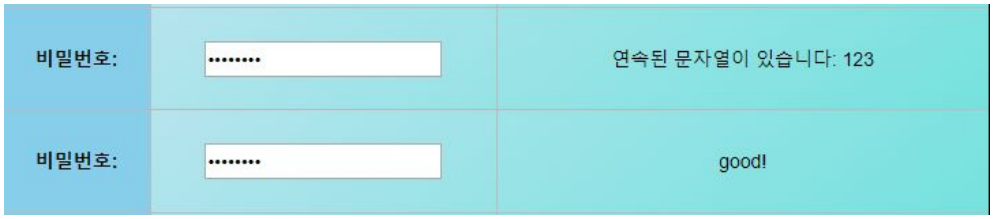
비밀번호의 유효성 검사 조건은 다음과 같다.
“6~10자 영문 대소문자, 숫자, 특수문자를 포함할 수 있다. 또한 동일한 숫자 및 문자를 3회 이상 사용할 수 없으며, 연속한 문자열(123, 321, abc, cba 등)을 사용할 수 없다.”
이 유효성 검사를 위해 작성한 JavaScript 코드는 다음과 같다.

```
169          var input2 = document.getElementById('userpwd');
170          input2.addEventListener('change', function(){ validation2(event.target.value); });
171          function validation2(value){//userpwd element의 값이 change됐을때 event처리
172            //userpwd의 길이(6~10자) 검사.
173            if (value.length<6 || value.length>10) {
174                document.getElementById('pwdval').innerHTML="비밀번호 길이는 6자~10자만 가능합니다. 현재길이 : " + value.length;
175                return false;
176            }
177
178            for (i = 0; i < value.length; i++) { //같은 문자가 3번이상 사용됐는지 검사
179
180              ch = value.charAt(i)
181              var cnt = 0;
182              for(j=0; j<value.length; j++) { // 비밀번호 전체에서 현재 문자가 몇번 사용됐는지를 카운트한다.
183                if(value.charAt(j) == ch){
184                  cnt++;
185                }
186              }
187              if(cnt>=3) // 카운트 횟수가 3회 이상(즉 같은문자가 3번이상 사용됨)이면 false를 리턴한다.
188              {
189                document.getElementById('pwdval').innerHTML="같은 문자나 숫자가 3회이상 반복됐습니다: " + ch;
190                return false;
191              }
192            }
193            for (i = 1; i < value.length-1; i++){  //연속된 문자열 존재여부 검사 (아스키코드의 차이로 검사함)
194              if((value.charAt(i)>='1' && value.charAt(i)<='8') || (value.charAt(i)>='B' && value.charAt(i)<='Y') || (value.charAt(i)>='b' && value.charAt(i)<='y'))
195              { // 이 범위를 만족하지 않으면 특수문자를 포함하고, 아스키코드 값의 차이조건을 만족하더라도 특수문자가 포함되면 연속된 문자열로 보지 않는다.
196                if(value.charCodeAt(i)-value.charCodeAt(i-1) == 1 && value.charCodeAt(i+1)-value.charCodeAt(i) == 1)
197                {// 아스키코드가 1차이로 연속됨 (ex. abc)
198                    document.getElementById('pwdval').innerHTML="연속된 문자열이 있습니다: " + value.charAt(i-1) + value.charAt(i) + value.charAt(i+1);
199                    return false;
200                }
201                if(value.charCodeAt(i)-value.charCodeAt(i-1) == -1 && value.charCodeAt(i+1)-value.charCodeAt(i) == -1)
202                {// 아스키코드가 -1차이로 연속됨 (ex. cba)
203                  document.getElementById('pwdval').innerHTML="연속된 문자열이 있습니다: " + value.charAt(i-1) + value.charAt(i) + value.charAt(i+1);
204                  return false;
205                }
206              }
207            }
208            document.getElementById('pwdval').innerHTML="good!";// 모든 조건 만족시 'good!' 메세지를 표시하며 유효성검사 통과
209            return true;
210          }
```

[비밀번호 유효성검사 JavaScript 코드] - register.html, line 169 ~ 210

validation2 함수는 비밀번호 입력란에 입력된 데이터를 parameter로 전달받는다. 사용자ID와 마찬가지로 6~10자의 길이제한이 있으므로, if문을 사용하여 길이검사를 하는것을 볼 수 있다. 영문

대소문자, 숫자, 특수문자를 포함할 수 있다는 조건에서 특수문자에 따로 제한사항이 없으므로 입력되는 데이터 문자에는 따로 제한이 없다. 다음 조건은 동일한 숫자 및 문자를 3회이상 사용할 수 없다는 조건이다. 이 조건을 검사하기 위해 2중 for문을 사용하였다. 길이가 최대 10자로 짧으므로, 입력된 모든 문자에 대해 전체에서 그 문자가 몇번 쓰였는지를 검사하고, 3회 이상일 경우 유효성 검사를 만족시키지 못하도록 한다. 마지막으로 연속된 문자열(123, 321, abc, cba 등)을 사용할 수 없다는 조건이다. 이 조건은 입력된 데이터를 처음부터 끝까지 조사하면서 연속으로 위치하는 모든 3자리 문자를 조사하는 방식으로 검사하였다. 문자열이 연속된다는 것은, 세 자리의 아스키코드 값의 차이가 각각 1이나 -1로 동일한 경우를 의미하므로 그것을 if문의 조건으로 지정하였다. 그리고 그 전에 if문을 하나 추가하여 “if((value.charAt(i)>='1' && value.charAt(i)<='8') || (value.charAt(i)>='B' && value.charAt(i)<='Y') || (value.charAt(i)>='b' && value.charAt(i)<='y'))” 이런 조건을 명시하는데 이것은 아스키코드상으로는 연속된 문자이더라도 특수문자인 경우는 연속한 문자로 보지 않기때문에 그런 경우를 따로 걸러내기 위한 조건이다. 예를들어, “>?@”는 아스키코드상으로는 62,63,64로 연속된 문자열이지만 이 과제에서는 이런 것들을 연속된 문자열로 보지 않는다. 따라서 이런 경우를 제외하고 숫자나 문자로 이루어져 있으면서 문자열이 연속된 경우만을 골라내 유효성검사를 만족시키지 않도록 한다.



[비밀번호 유효성검사 웹페이지 동작] (qwert123/qwert124)

- **생년월일**

생년월일의 유효성 검사 조건은 따로 명시되어 있지 않다. 그래서 일반적인 생년월일 검사 조건을 사용하였다. 입력 형식은 ‘yyyy-mm-dd’와 같은 형식으로만 입력할 수 있도록 하였고, 1800년대나 미래의 시간과 같이 현재 생년월일로 사용될 수 없는 값들은 유효성검사를 만족하지 못하도록 하기 위해 regular expression을 사용하였다. 사용한 regular expression은 다음과 같다.

“new RegExp(/^(19|20)\d{2}-(0[1-9]|1[012])-(0[1-9]|[12][0-9]|3[0-1])$/);”

위에서 설명한 생년월일의 유효성 검사 조건은 위의 regular expression으로 표현할 수 있다.
이 유효성 검사를 수행하기 위해 구현한 validation3 함수는 생년월일 입력 폼의 데이터를 parameter로 전달받는다.
전체 JavaScript 코드는 다음과 같다.

```
211  var input3 = document.getElementById('birthday');
212  input3.addEventListener('change', function(){ validation3(event.target.value); });
213  function validation3(value){
214    var pattern=new RegExp(/^(19|20)\d{2}-(0[1-9]|1[012])-(0[1-9]|[12][0-9]|3[0-1])$/);
215    //생년월일은 regular expression을 사용하여 유효성검사 함.
216    if(!pattern.test(value)) // 지정해준 regular expression을 만족하는지 여부를 검사함.
217    {
218      document.getElementById('birthval').innerHTML="생년월일의 형식이나 값을 확인해주세요.";
219      return false;
220    }
221    document.getElementById('birthval').innerHTML="good!";
222    return true;
223  }
```

[생년월일 유효성검사 JavaScript 코드] - register.html, line 211 ~ 223

- **휴대전화번호**

휴대전화번호의 유효성 검사 조건은 따로 명시되어 있지 않다. 그래서 일반적인 휴대전화번호 검사 조건을 사용하였다. 입력 형식은 '###-####-####'와 같은 형식으로만 입력할 수 있고, 가운데 4자리는 3자리로 입력할 수도 있도록 하였다. 이런 조건을 구현하기 위해 regular expression을 사용하였다. 사용한 regular expression은 다음과 같다.

"new RegExp(/^\d{3}-\d{3,4}-\d{4}$/);"

이 유효성 검사를 수행하기 위해 구현한 validation4 함수는 휴대전화번호 입력 폼의 데이터를 parameter로 전달받는다.
전체 JavaScript 코드는 다음과 같다.

```
224  var input4 = document.getElementById('usertel');
225  input4.addEventListener('change', function(){ validation4(event.target.value); });
226  function validation4(value){
227    var pattern=new RegExp(/^\d{3}-\d{3,4}-\d{4}$/);
228    //휴대전화번호는 regular expression을 사용하여 유효성검사 함.
229    if(!pattern.test(value)) // 지정해준 regular expression을 만족하는지 여부를 검사함.
230    {
231      document.getElementById('telval').innerHTML="휴대전화번호 형식을 확인하세요";
232      return false;
233    }
234    document.getElementById('telval').innerHTML="good!";
235    return true;
236  }
```

[휴대전화번호 유효성검사 JavaScript 코드] - register.html, line 224 ~ 236

여기까지가 과제에서 기본적으로 요구하는 입력 데이터들의 유효성 검사이다. 다음은 "추가로 한 개 이상의 입력 데이터 (주소 등) 의 유효성을 검사한다." 는 과제의 요구조건을 위해 추가한 E-mail 입력 데이터에 대한 유효성 검사를 설명한다.

- **E-mail 주소**

과제에서 요구하는 Form 입력 데이터 외에 추가적으로 E-mail 입력 데이터 형식을 추가하였다. E-mail 주소에 대한 유효성 검사는 역시 regular expression을 사용하였다. 입력되는 데이터는 "(알파벳,숫자,_,.,-)@(알파벳,숫자,-).(알파벳,숫자,-)" 와 같은 형식으로 입력되어야 하고, 그 조건을 regular expression으로 구현하였다. 사용한 regular expression은 다음과 같다.

"new RegExp(/^[A-Za-z0-9_\.\-]+@[A-Za-z0-9\-]+\.[A-Za-z0-9\-]+/);"

이 유효성 검사를 수행하기 위해 구현한 validation5 함수는 E-mail 입력 폼의 데이터를 parameter로 전달받는다.
전체 JavaScript 코드는 다음과 같다.

```
237  var input5 = document.getElementById('useremail'); //useremail element를 HTML DOM API를 사용하여 input4변수에 assign
238  input5.addEventListener('change', function(){ validation5(event.target.value); });
239  function validation5(value){
240    var pattern=new RegExp(/^[A-Za-z0-9_\.\-]+@[A-Za-z0-9\-]+\.[A-Za-z0-9\-]+/);
241    //E-mail주소는 regular expression을 사용하여 유효성검사 함.
242    if(!pattern.test(value)) // 지정해준 regular expression을 만족하는지 여부를 검사함.
243    {
244      document.getElementById('emailval').innerHTML="메일주소 형식을 확인하세요";
245      return false;
246    }
247    document.getElementById('emailval').innerHTML="good!";
248    return true;
249  }
```

[E-mail 주소 유효성검사 JavaScript 코드] - register.html, line 237 ~ 249

이렇게 모든 입력 값들에 대한 유효성 검사가 끝나면, 회원가입 Form이 들어있는 테이블의 가장 오른쪽 Column에 전부 "good!" 이라는 메시지가 나타난다. 유효성 검사를 전부 만족시켜 완료된 후 회원가입 버튼을 누르면 서버에서 받은 response 메시지가 뜨고, 유효성 검사를 만족시키지 못한 상태에서 회원가입 버튼을 누르면 적절하지 않은 값이 있다는 메시지가 alert된다.

2. 유효성 검사 완료 이후 요구사항

- 유효성 검사가 완료된 후 회원가입 버튼을 누르면 http://165.132.105.212/~icl/project2.php 에 Ajax를 통해서 회원가입 정보를 POST request로 보내고 그 결과로 받은 response를 사용자에게 띄워준다
- http://165.132.105.212/~icl/project2.php 에 request는 JSON 형식으로 보내야 하며, 'id', 'pw'라는 이름으로 아이디와 비밀번호 정보가 포함되어야 한다.

위 기능을 구현하기 위해 Ajax 기술을 사용하여 서버로 POST request를 보냈고, 그 때 JSON 기술을 사용하여 '사용자ID'와 '비밀번호'를 포함하는 Form 데이터의 일부를 전송하여 서버사이드에 한번 더 회원가입 데이터를 검증할 수 있도록 한다. 그리고 그에 맞는 response를 받아서 페이지에 비동기적으로 나타날 수 있도록 구현하였다. 각 기술을 구현한 코드에 대한 명시는 뒤에 이어진다. 여기서는 각 요구사항에 대한 웹페이지 실행 화면을 보이고 있다.

[서버로부터 받은 response 비동기적으로 표시]

3. 요구 사항 명시(구현한 기술에 대한 설명)

a. HTML/XML DOM API

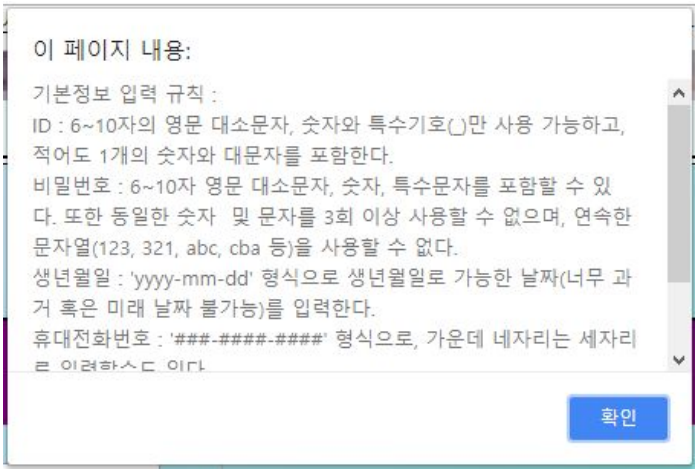
HTML Document에 작성된 Form에서 각 input 엘리먼트들에 Access하기 위해 HTML DOM API를 사용한다.

```
var input1 = document.getElementById('userid');
```

[HTML DOM API] - register.html, line 129

document.getElementById를 이용해서 id='userid'인 엘리먼트에 Access

다음으로 XML DOM API를 사용하는 부분이다. 다음 그림과 같이 '유효성 검사조건'이라는 버튼을 누르면 사용자가 회원가입 데이터에 적용되는 유효성 검사 조건을 확인할 수 있도록 하였다. 이 때, XML을 이용해서 유효성 검사조건 데이터를 store 및 transport하고 있으며, XML로의 데이터에 Access하여 그 값을 가져오기 위해 XML DOM API를 사용하고 있다.



['유효성 검사조건' 버튼을 눌렀을 때 확인 가능한 상세조건]

```
<button type="button" onclick="rulecheck()">유효성 검사조건</button>
```

```
function rulecheck(){
  var parser, xmlDoc;
  var text = "<result>"+
      "<id>ID : 6~10자의 영문 대소문자, 숫자와 특수기호(_)만 사용 가능하고, 적어도 1개의 숫자와 대문자를 포함한다.</id>" +
      "<pwd>비밀번호 : 6~10자 영문 대소문자, 숫자, 특수문자를 포함할 수 있다. 또한 동일한 숫자  및 문자를 " +
      "3회 이상 사용할 수 없으며, 연속한 문자열(123, 321, abc, cba 등)을 사용할 수 없다. </pwd>" +
      "<birth>생년월일 : 'yyyy-mm-dd' 형식으로 생년월일로 가능한 날짜(너무 과거 혹은 미래 날짜 불가능)를 입력한다.</birth>" +
      "<tel>휴대전화번호 : '###-####-####' 형식으로, 가운데 네자리는 세자리로 입력할수도 있다.</tel>" +
      "<email>E-mail : '(알파벳,숫자,_,.,-)@(알파벳,숫자,-).(알파벳,숫자,-)' 형식으로 '@'와 '.'을 포함한다.</email>" +
    "</result>"; //XML로 parsing하기 위한 xml string을 생성중. 사용자가 유효성검사 규칙을 확인할 수 있도록 그 내용을 담고있다.

  parser = new DOMParser(); // xml을 parsing하기 위해 parser생성
  xmlDoc = parser.parseFromString(text,"text/xml"); // xml string을 xml document로 parsing

  var rule = "기본정보 입력 규칙 :\n" ;
  rule += xmlDoc.getElementsByTagName("id")[0].childNodes[0].nodeValue + "\n";
  rule += xmlDoc.getElementsByTagName("pwd")[0].childNodes[0].nodeValue + "\n";
  rule += xmlDoc.getElementsByTagName("birth")[0].childNodes[0].nodeValue + "\n";
  rule += xmlDoc.getElementsByTagName("tel")[0].childNodes[0].nodeValue + "\n";
  rule += xmlDoc.getElementsByTagName("email")[0].childNodes[0].nodeValue + "\n";
  //XML DOM API를 사용하여 xml에 저장된 데이터를 가져오고 rule변수에 저장하고 있음.
  alert(rule);
}
```

[XML DOM API] - register.html, line 282 ~ 304

b. JavaScript event

JavaScript event는 입력 데이터의 유효성검사 전반에 걸쳐 많이 사용되고 있다. 그 중에 한 곳을 다음과 같이 명시한다.

```
131             input1.addEventListener('change', function(){ validation1(event.target.value); }); // 요구기술 : JavaScript event
132             function validation1(value){//userid element의 값이 change됐을때 event처리
```

[JavaScript event] - register.html, line 131

JavaScript event를 사용하여 Form의 각 input데이터에 변경이 있을 때 특정 함수가 실행되도록 event 처리를 해주고 있다.

c. AJAX

d. JSON

위에서 설명한 것 처럼, 입력 데이터에 대한 유효성 검사가 완료된 후 '회원가입' 버튼을 누르면 AJAX기술을 사용하여 서버로 request를 보내고 그에 맞는 response를 받아 비동기적으로 웹페이지에 표시하는 기능을 구현하였다. 서버로 보내는 데이터를 JSON 형식을 사용하였다. JSON Object를 생성하여 각각의 입력데이터들로 key와 value pair들을 저장하고, 서버에 전송할때는 JSON.stringify() 함수를 사용하여 JSON을 string으로 바꿔주었다.

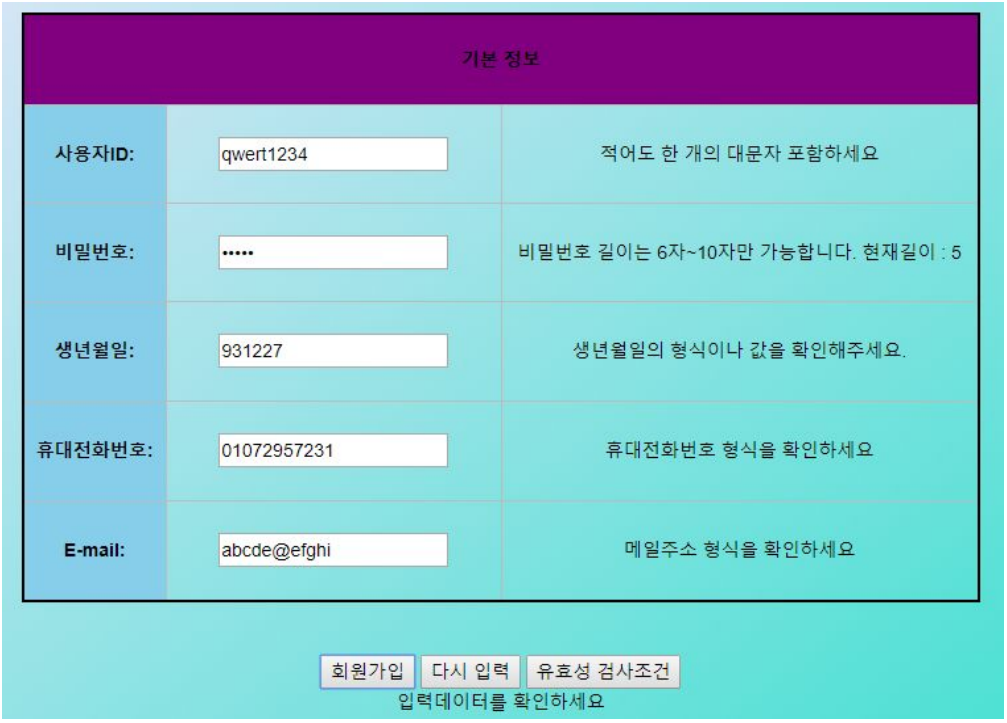
```
121                     <button type="button" onclick="myfunc()">회원가입</button>
```

```
252         function myfunc(){
253           if(validation1(registerform.userid.value) && validation2(registerform.userpwd.value) &&
254           validation3(registerform.birthday.value) && validation4(registerform.usertel.value) && validation5(registerform.useremail.value)){
255             // 모든 form 입력값들이 유효성 검사를 통과함
256             var xhr = new XMLHttpRequest(); // 요구기술 : AJAX
257             xhr.open("POST", "http://165.132.105.212/~icl/project2.php"); // request를 보낼 URL과 방식(POST)를 지정함
258             xhr.onreadystatechange = function(){ // readyState property가 변경되었을 때 호출될 함수를 정의함
259               if(xhr.readyState === 4 && xhr.status === 200){ // readyState가 4라는 것은 request가 finish되고 response가 준비됨을 의미함
260                 // status가 200이라는 것은 request가 잘 처리되었음을 의미함.
261                 document.getElementById("response").innerHTML = xhr.responseText; //response라는 <p>엘리먼트 안에 responseText를 표시하도록 설정함
262               }
263             }
264             xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
265
266             var obj = new Object(); // JSON object를 생성함
267             obj.id = registerform.userid.value;
268             obj.pw = registerform.userpwd.value;
269             obj.birth = registerform.birthday.value;
270             obj.email = registerform.useremail.value;
271             obj.tel = registerform.usertel.value; // object의 key와 value pair들을 assign함.
272             var json_data = JSON.stringify(obj); // JSON을 이용하여 서버에 데이터를 전송하기 위해 string으로 변환해주는 stringify 함수를 사용하고 있음.
273             var STRING_DATA = 'data='+json_data; // 서버에서 요구하는 데이터 형식을 맞춰주고 있음.
274             // 요구기술 : JSON
275             xhr.send(STRING_DATA);
276           }
277           else { // 유효성검사조건을 하나라도 만족하지 않은 상태에서 회원가입 버튼을 누르면 클라이언트 사이드에서 alert메시지를 표시해 다시 입력하도록 함.
278             document.getElementById("response").innerHTML = "입력데이터를 확인하세요";
279             alert("적절하지 않은 입력값이 있습니다.");
280           }
281         }
```

[AJAX] - register.html, line 252 ~ 281

[JSON] - register.html, line 266 ~ 273

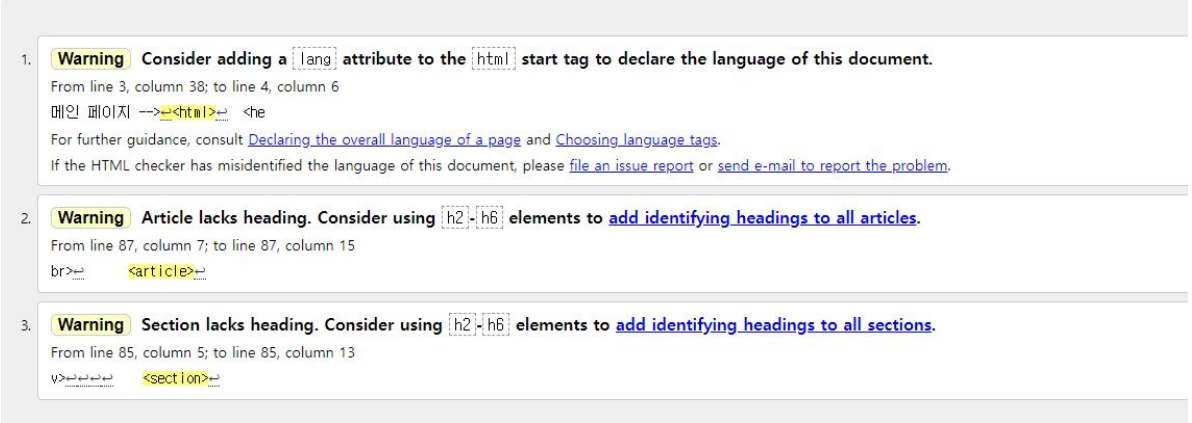
4. 웹페이지 동작 화면



유효성 검사를 만족하지 못하는 경우

유효성 검사를 만족한 후, 회원가입 버튼을 누른 경우

5. Validation



register.html 에 대한 Validation 화면