JS201
Javascript 중급 응용

```
/**
 * @name 백창열
 * @email cybaek@netsgo.com
 * @homepage http://cybaek.com/
 */
```

목 차

<form> 검증 자동화 함수 제작

<form>값에 데이터를 검증하기 위해 수업이 많은 검증 함수가 매 <form>마다 따라 다닌다. 이러한 단순 작업은 개발하는 사람도 귀찮고 간혹 실수를 유발하기도 한다. 단순 작업은 짜증의 대상이 아니라, 자동화의 대상이 되기에 적절하므로 반가워 해야한다.

이 장의 내용을 이해하기 위해선 Javascript와 <form>태그에 대한 이해가 꼭 필요하다.

학습 목표

- ❑ <form>값 검증하는 함수를 제작한다.
- ❑ 함수는 각 태그의 속성에 기반하여 동작하게 하고 이 함수를 사용하는 개발자가 함수 작성을 최소화 하도록 돕는다.

연습문제: <form>의 모든 내용 읽기

내용

- ❑ <form>의 모든 elements의 value 속성을 alert()로 찍는다.
- ❑ 제한 시간: 15분

연습문제 권장코드: <form>의 모든 내용 읽기

```
validation.1.html

<head>
<script>
function checkValue(f){
    var j = f.elements.length
    var i;
    var re;
    var curr;

    for (i=0; i<j; i++)
    {
            curr = f.elements[i];

            alert(curr.value);
    }

    return true;
}
</script>
</head>
<body>
<form onsubmit="return checkValue(this);">
Name : <input type=text name=name><br>
Age : <input type=text name=age><br>
```

```
<input type=submit>
</form>

</body>
```

이 코드를 수행하면 정말 모든 객체의 값을 출력한다. (생각지도 않았던 submit 버튼도 출력되었다.)

연습문제: 검증대상선별

<form>에는 많은 객체가 있지만 모든 객체의 값을 검증하진 않는다. 이번에는 검증대상을 선별하도록 하자.

내용

- ❏ 검증이 필요한 객체에는 V란 속성을 주고 그 값을 true로 셋팅한다.
- ❏ checkValue에서 V=true인 것만 값과 그 값의 길이를 출력한다.
- ❏ 제한시간: 10분

연습문제 권장코드: 검증대상선별

```
validation.2.html

<head>
<script>
function checkValue(f){
    var j = f.elements.length
    var i;
    var re;
    var curr;

    for (i=0; i<j; i++)
    {
            curr = f.elements[i];

            if (typeof(curr.V) == 'undefined') continue;
            if (curr.V != 'true') continue;

            alert(curr.value + ', ' + curr.value.length);
    }

    return true;
}
</script>
</head>
<body>
<form onsubmit="return checkValue(this);">
Name : <input type=text name=name V=true><br>
Age : <input type=text name=age V=true><br>
<input type=submit>
</form>

</body>
```

한글을 입력하고 길이를 보면 한글 한 자를 길이 1로 반환하는 것을 알 수 있다. 그러나, DB에서 한글 한 자는 2바이트를 차지하기 때문에 이 값으로 길이를 검증할 수 없다.

다음 코드는 한글 1자의 길이를 2로 반환하는 b_length()라는 함수를 정의하고 그것으로 대체했다.

```
validation.3.html

<head>
<script>
function checkValue(f){
    var j = f.elements.length
    var i;
    var re;
    var curr;

    for (i=0; i<j; i++)
    {
            curr = f.elements[i];

            if (typeof(curr.V) == 'undefined') continue;
            if (curr.V != 'true') continue;

            alert(curr.value + ', ' + b_length(curr.value));
    }

    return true;
}

function b_length(str){
      var iRet = 0;
      var iLen = str.length;

      for (i=0; i<iLen; i++){
            if ((str.charCodeAt(i)<0) || (str.charCodeAt(i)>127)){
                  iRet = iRet + 1;
            }
      }

      return (iLen + iRet);
}

</script>
</head>
<body>
<form onsubmit="return checkValue(this);">
Name : <input type=text name=name V=true><br>
Age : <input type=text name=age V=true><br>
<input type=submit>
</form>

</body>
```

문자열의 길이 검증

<input> 객체에 MaxLen과 MinLen이란 속성이 없지만 HTML에 이 속성을 적고 값을 명시하면 스크립트를 통해 그 값을 읽을 수 있다. 이러한 특성을 이용하여 <form>값 검증을 자동화 할 수 있다.

다음 코드는 MaxLen, MinLen 값으로 길이를 검증한다.

```
validation.4.html

<head>
<script>
function checkValue(f){
    var j = f.elements.length
    var i;
    var re;
    var curr;

    for (i=0; i<j; i++)
    {
            curr = f.elements[i];

            if (typeof(curr.V) == 'undefined') continue;
            if (curr.V != 'true') continue;

            if (!checkLength(curr)){
                  return false;
            }
    }

    return true;
}

function checkLength(curr){
      var checkLen = true;
      var minLen, maxLen, length;

      minLen = parseInt(curr.MinLen);
      maxLen = parseInt(curr.MaxLen);

      length = parseInt(b_length(curr.value));
      if ((length<minLen) || (length>maxLen)){
            curr.focus();
            alert('길이가 올바르지 않습니다.');
            return false;
      }

      return true;
}

function b_length(str){
      var iRet = 0;
      var iLen = str.length;

      for (i=0; i<iLen; i++){
            if ((str.charCodeAt(i)<0) || (str.charCodeAt(i)>127)){
                  iRet = iRet + 1;
            }
      }

      return (iLen + iRet);
}

</script>
</head>
<body>
<form onsubmit="return checkValue(this);">
Name : <input type=text name=name
      V=true
      MinLen=6
      MaxLen=10><br>
Age : <input type=text name=age
      V=true
      MinLen=1
      MaxLen=3><br>
<input type=submit>
</form>

</body>
```

MinLen, MaxLen 뿐만 아니라 앞으로 여러가지 속성을 더 정의할 것이다. 따라서 이 필드는 필수 필드가 아니기에 위에 코드는 잠재적인 위험을 내포하고 있다.

다음 코드는 두 필드가 정의되어 있는 경우만 길이 검증을 수행하고, 오류메시지를 사용자가 정의하도록 한다. 만일 오류메시지를 정의하지 않은 경우는 기본 메시지를 사용한다.

```
validation.5.html

var ERRMSG_DEFAULT_LEN = “입력한 값의 길이가 적당하지 않습니다.”
...

function checkLength(curr){
      var checkLen = true;
      var minLen, maxLen, length;
      minLen = parseInt(curr.MinLen);
      if (isNaN(minLen)){
            return true;
      }
      maxLen = parseInt(curr.MaxLen);
      if (isNaN(maxLen)){
            return true;
      }

      length = parseInt(b_length(curr.value));
      if ((length<minLen) || (length>maxLen)){
            curr.focus();
            if (typeof(curr.LenErrMsg) == "undefined"){
                  curr.LenErrMsg = ERRMSG_DEFAULT_LEN;
            }
            var errMsg = curr.LenErrMsg;
            errMsg = errMsg.replace("$MinLen", curr.MinLen);
            errMsg = errMsg.replace("$MaxLen", curr.MaxLen);
            alert(errMsg);
            return false;
      }

      return true;
}

...
```

전체 코드

이 모듈은 태그만으로 값의 길이, 숫자로 된 값의 범위를 검증할 수 있고, 보다 정교한 검증을 위해 정규표현을 사용할 수 있다. 이런 것만으로 힘들 때는 사용자가 정의한 Javascript함수를 이용하여 검증할 수 있다. 더불어 좌우 양 끝의 공백 제거와 금지문자 검출 기능도 갖고 있다.

전체 코드는 다음과 같다.

```
cybaek_validation.js

/*
 * Validation Library
 *
 * @version 1.1
 * @author BAEK, CHANG YOL http://cybaek.com/
 * @date 2003.11.26
 */
var ERRMSG_DEFAULT_SCOPE = "값의 범위가 적당하지 않습니다. (최소: $MinScope, 최대: $MaxScope)";
var ERRMSG_DEFAULT_LEN = "입력한 값의 길이가 적당하지 않습니다. (최소: $MinLen, 최대: $MaxLen)";
var ERRMSG_DEFAULT_CHECKFUNC = "입력값이 올바르지 않습니다.";
var ERRMSG_DEFAULT_FORBIDDEN_CHARS = "$ForbiddenChars는 금지 문자이고, $FoundChar가 발견되었습니다."
var ERRMSG_DEFAULT_REGEXP = "입력값이 올바르지 않습니다."
var ERRMSG_NUMBER = "숫자를 입력하십시오.";

function checkValue(f){
    var j = f.elements.length
    var i;
    var re;
    var curr;

    for (i=0; i<j; i++)
    {
            curr = f.elements[i];
        if (typeof(curr.V) == "undefined") continue;

            preprocess(curr);

            if (!checkLength(curr) ||
                  !checkScope(curr) ||
                  !checkWithFunc(curr) ||
                  !checkWithRegExp(curr)){
                  return false;
            }
            if (!checkForbiddenChars(curr)){
                  return false;
            }
    }

    return true;
}

function preprocess(curr){
      removeExtremeSpace(curr);
}

function removeExtremeSpace(curr){
      if (typeof(curr.Trim) != "undefined"){
            if (curr.Trim.indexOf("L") != -1){
                  curr.value = ltrim(curr.value);
            }
            if (curr.Trim.indexOf("R") != -1){
                  curr.value = rtrim(curr.value);
            }
      }
}

// @author: brad@vermontsoftware.com
function ltrim(str){
   var whitespace = new String(" \t\n\r");

   var s = new String(str);

   if (whitespace.indexOf(s.charAt(0)) != -1) {
      // We have a string with leading blank(s)...

      var j=0, i = s.length;

      // Iterate from the far left of string until we
      // don't have any more whitespace...
      while (j < i && whitespace.indexOf(s.charAt(j)) != -1)
         j++;

      // Get the substring from the first non-whitespace
      // character to the end of the string...
      s = s.substring(j, i);
   }
   return s;
}

// @author: brad@vermontsoftware.com
function rtrim(str){
   // We don't want to trip JUST spaces, but also tabs,
   // line feeds, etc.  Add anything else you want to
   // "trim" here in Whitespace
   var whitespace = new String(" \t\n\r");

   var s = new String(str);

   if (whitespace.indexOf(s.charAt(s.length-1)) != -1) {
      // We have a string with trailing blank(s)...
      var i = s.length - 1;       // Get length of string

      // Iterate from the far right of string until we
      // don't have any more whitespace...
      while (i >= 0 && whitespace.indexOf(s.charAt(i)) != -1)
         i--;


      // Get the substring from the front of the string to
      // where the last non-whitespace character is...
      s = s.substring(0, i+1);
   }

   return s;
}

// @author: brad@vermontsoftware.com
function trim(str){
   return rtrim(ltrim(str));
}


function checkLength(curr){
      var checkLen = true;
      var minLen, maxLen, length;
      minLen = parseInt(curr.MinLen);
      if (isNaN(minLen)){
            return true;
      }
      maxLen = parseInt(curr.MaxLen);
      if (isNaN(maxLen)){
            return true;
      }

      length = parseInt(b_length(curr.value));
      if ((length<minLen) || (length>maxLen)){
            curr.focus();
            if (typeof(curr.LenErrMsg) == "undefined"){
                  curr.LenErrMsg = ERRMSG_DEFAULT_LEN;
            }
            var errMsg = curr.LenErrMsg;
            errMsg = errMsg.replace("$MinLen", curr.MinLen);
            errMsg = errMsg.replace("$MaxLen", curr.MaxLen);
            alert(errMsg);
            return false;
      }

      return true;
}

function checkScope(curr){
      var checkScope = true;
      var minScope, maxScope;

      minScope = parseInt(curr.MinScope);
      if (isNaN(minScope)){
            return true;
      }
      maxScope = parseInt(curr.MaxScope);
      if (isNaN(maxScope)){
            return true;
      }

      if (isNaN(curr.value)){
            curr.focus();
            alert(ERRMSG_NUMBER);
            return false;
      }

      if ((curr.value > maxScope) || (curr.value < minScope)){
            curr.focus();
            if (typeof(curr.ScopeErrMsg) == "undefined"){
                  curr.ScopeErrMsg = ERRMSG_DEFAULT_SCOPE;
            }
            var errMsg = curr.ScopeErrMsg;
            errMsg = errMsg.replace("$MinScope", curr.MinScope);
            errMsg = errMsg.replace("$MaxScope", curr.MaxScope);
            alert(errMsg);
            return false;
      }

      return true;
}

function checkWithFunc(curr){
      var result;

      if (typeof(curr.CheckFunc) != "undefined"){
            try{
                  result = eval(curr.CheckFunc+"(curr, curr.value);");
            }
            catch(e){
                  alert("입력값 검사 함수 " + curr.CheckFunc + "(src, value) 에 오류가
있습니다.");
                  result = false;
            }
            if (!result){
                  curr.focus();
                  if (typeof(curr.CheckFuncErrMsg) == "undefined"){
                        curr.CheckFuncErrMsg = ERRMSG_DEFAULT_CHECKFUNC;
                  }
                  alert(curr.CheckFuncErrMsg);
                  return false;
            }
      }
      return true;
}

function checkForbiddenChars(curr){
      if (typeof(curr.ForbiddenChars) == "undefined"){
            return true;
      }

      var length, i, currChar, value;
      length = curr.value.length;
      value = curr.value;
      for(i=0; i<length; i++){
            currChar = value.charAt(i);
            if (curr.ForbiddenChars.indexOf(currChar) != -1){
                  curr.focus();
                  if (typeof(curr.ForbiddenCharsErrMsg) == "undefined"){
                        curr.ForbiddenCharsErrMsg = ERRMSG_DEFAULT_FORBIDDEN_CHARS;
                  }
                  var errMsg = curr.ForbiddenCharsErrMsg;
                  errMsg = errMsg.replace("$ForbiddenChars", curr.ForbiddenChars);
                  errMsg = errMsg.replace("$FoundChar", currChar);
                  alert(errMsg);
                  return false;
            }
      }
      return true;
}

function checkWithRegExp(curr){
      if (typeof(curr.RegExp) == "undefined"){
            return true;
      }

      var re = new RegExp(curr.RegExp, "gi");
      if (!re.test(curr.value)){
            curr.focus();
            if (typeof(curr.RegExpErrMsg) == "undefined"){
                  curr.RegExpErrMsg = ERRMSG_DEFAULT_REGEXP;
            }
            var errMsg = curr.RegExpErrMsg;
            errMsg = errMsg.replace("$RegExp", curr.RegExpErrMsg);
            alert(errMsg);
            return false;
      }

      return true;
}

function b_length(str){
      var iRet = 0;
      var iLen = str.length;

      for (i=0; i<iLen; i++){
            if ((str.charCodeAt(i)<0) || (str.charCodeAt(i)>127)){
                  iRet = iRet + 1;
            }
      }

      return (iLen + iRet);
}
```

```
sample.html

<!--
값의 길이를 검증하는 보기입니다.
이 라이브러리를 이용하여 값을 검증하고 싶을 땐 해당 객체에 반드시 V 속성을 정의해야합니다.
-->
```

```
<script src=cybaek_validation.js></script>

<form onsubmit="return checkValue(this);">
<!--
* 길이 검증

필수 필드
V, MinLen, MaxLen

선택 필드
LenErrMsg: 이 속성을 정의하지 않으면 js에 있는 기본 메시지로 출력됩니다.
오류메시지에 $MinLen, $MaxLen 변수를 혼합하여 메시지를 만들 수 있습니다.
-->
Name : <input type=text name=name
                  V=true
                  MinLen="6" MaxLen="10"
                  LenErrMsg="길이가 최소 $MinLen자, 최대 $MaxLen자 이어야 합니다."
                  >

<!--
* 숫자 범위 검증

필수 필드
V, MinScope, MaxScope

선택 필드
ScopeErrMsg: 이 속성을 정의하지 않으면 js에 있는 기본 메시지로 출력됩니다.
오류메시지에 $MinScope, $MaxScope 변수를 혼합하여 메시지를 만들 수 있습니다.
-->
Age : <input type=text name=age
                  V=true
                  MinScope="14" MaxScope="100"
                  ScopeErrMsg="값의 범위가 적당하지 않습니다. $MinScope보다 크거나 같고
$MaxScope보다 작거나 같아야 합니다."
                  >


<!--
* 임의 함수를 이용한 값 검증

필수 필드
V, CheckFunc

선택 필드
CheckFuncErrMsg: 이 속성을 정의하지 않으면 js에 있는 기본 메시지로 출력됩니다.
-->
Zip : <input type=text name=zip
                  V=true
                  MinLen="7"  MaxLen="7"
                  LenErrMsg="길이가 적당하지 않습니다."
                  CheckFunc="checkZip"
                  CheckFuncErrMsg="올바른 우편번호가 아닙니다."
                  Trim="RL"
                  >
                  <br>
```

```
<!--
* 금지 문자 검출

필수 필드
V, ForbiddenChars

선택 필드
ForbiddenCharsErrMsg: 이 속성을 정의하지 않으면 js에 있는 기본 메시지로 출력됩니다.
-->
Gender : <input type=text name=gender
                  V=true
                  ForbiddenChars="!@#$%^&*"
                  ForbiddenCharsErrMsg="$ForbiddenChars는 금지 문자이고, $FoundChar가
발견되었습니다."
                  >

<!--
* 정규표현 적용

필수 필드
V, RegExp

선택 필드
RegExpErrMsg 이 속성을 정의하지 않으면 js에 있는 기본 메시지로 출력됩니다.
-->
학과목코드: <input type=text name=code
                  V=true
                  RegExp="^[a-zA-Z]{2}\d\d\d$"
                  RegExpErrMsg="값의 형식이 올바르지 않습니다. (예, CS101, CE502)"
                  >

<input type=submit>
</form>


<script>
function checkZip(src, value){
      alert('값: ' + value);
    return true;
}

</script>
```