

6장. 유효성 검사



Validation / Regular Expression



유효성 검사(validation)

유효성 검사(validation)

- 사용자가 폼 페이지에서 입력한 데이터 값이 서버로 전송되기 전에 특정 규칙에 맞게 입력되었는지 검증하는 것을 말한다.

예를 들면, 회원 가입시 아이디나 비밀번호, 연락처등 검사, 아이디 중복 검사등을 들 수 있다.

유효성 검사를 위한 핸들러 함수

- 폼 페이지에서 이벤트가 발생했을때(<input onclick>을 클릭한 경우)의 유효성 검사를 위한 **매핑 메서드로 자바스크립트**를 이용하여 코드를 작성한다.

```
<script type="text/javascript">  
    function 핸들러 함수(){  
        var str = document.폼 이름.입력항목 이름.value;  
    }  
</script>  
  
<form name="폼 이름">  
    <input type="button" onclick="핸들러함수()">  
</form>
```

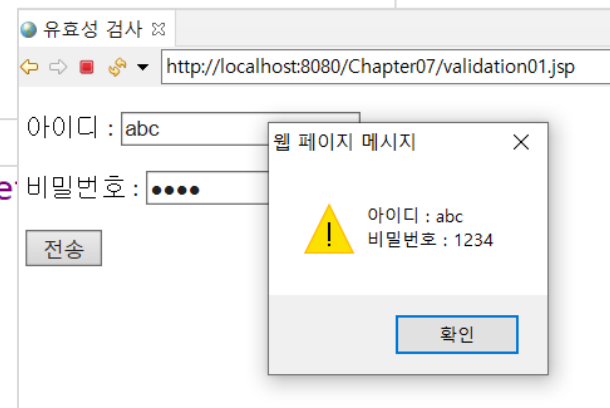


유효성 검사(validation)

폼 페이지에 입력한 아이디와 비밀번호 출력하기

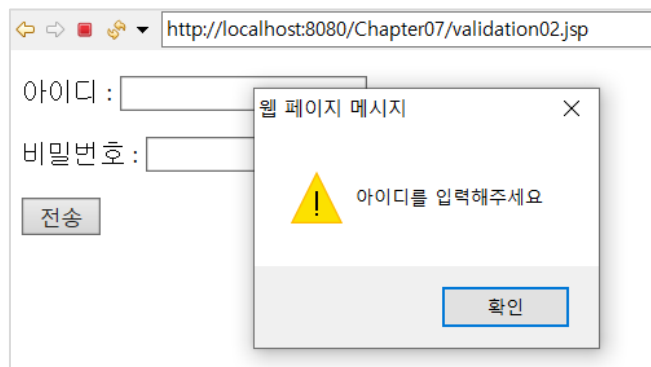
```
<title>유효성 검사</title>
</head>
<script type="text/javascript">
    function checkForm(){
        alert("아이디 : " + document.loginForm.userid.value + "\n"
            + "비밀번호 : " + document.loginForm.passwd.value)
    }
</script>
```

```
<form name="LoginForm" action="LoginProcess.jsp" method="post">
    <p>
        <label for="id">아이디 : </label>
        <input type="text" id="id" name="userid">
    </p>
    <p>
        <label for="pwd">비밀번호 : </label>
        <input type="password" id="pwd" name="passwd">
    </p>
    <p><input type="submit" value="전송" onclick="checkForm()"></p>
</form>
```



유효성 검사(validation)

폼 페이지에 입력한 데이터 값의 유/무 검사하기 - [기본 유효성 검사]



※ return false 생략하면
action 페이지로 이동함

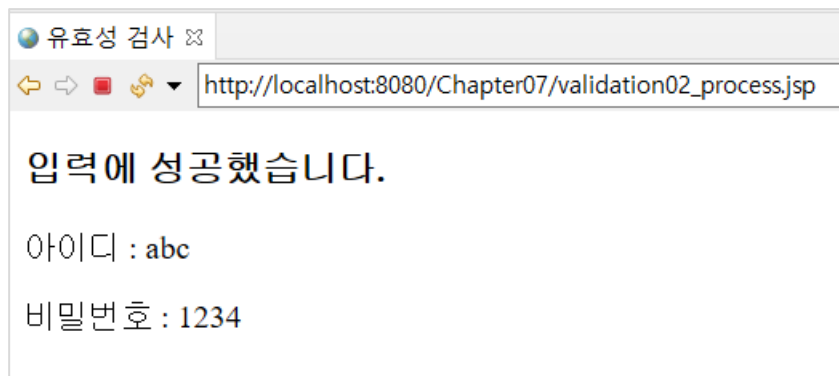
```
<script type="text/javascript">
    function checkLogin(){
        var form = document.loginForm;
        if(form.id.value == ""){
            alert("아이디를 입력해주세요");
            form.id.focus();
            return false;
        }else if(form.pwd.value == ""){
            alert("비밀번호를 입력해주세요");
            form.pwd.focus();
            return false;
        }else{
            form.submit();
        }
    }
</script>
```



유효성 검사(validation)

폼 페이지에 입력한 데이터 값의 유/무 검사하기 - [기본 유효성 검사]

```
<h3>입력에 성공했습니다.</h3>
<%
    String id = request.getParameter("userid");
    String pwd = request.getParameter("passwd");
%>
<p>아이디 : <%=id %>
<p>비밀번호 : <%=pwd %>
```



유효성 검사(validation)

폼 페이지에 입력한 데이터 값의 유/무 검사하기

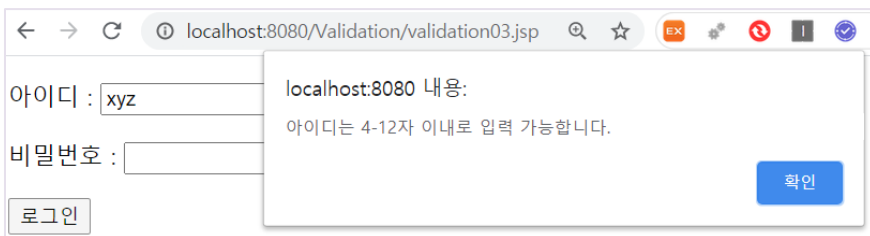
```
Sources  Network  Performance  Memory  Application
validation02.jsp x
12      alert("아이디를 입력해주세요");
13      form.id.focus();
14      return false;
15      }else if(form.pwd.value == ""){
16      alert("비밀번호를 입력해주세요");
17      form.pw.focus(); x
18      return false;
19      }else{
20      form.submit();
```

브라우저에서 오류
확인

유효성 검사(validation)

폼 페이지에 입력한 데이터 길이 확인하기

document.폼 이름.입력양식 이름.value.length



A screenshot of a web browser window at localhost:8080/Validation/validation03.jsp. The browser shows a login form with two input fields: '아이디 : xyz' and '비밀번호 :'. Below the password field is a '로그인' button. An alert box is displayed over the form, titled 'localhost:8080 내용:', with the message '아이디는 4-12자 이내로 입력 가능합니다.' and a '확인' button.

```
function checkLogin(){
    var form = document.loginForm;

    if(form.userid.value.length < 4 || form.userid.value.length > 12){
        alert("아이디는 4~12자 이내로 입력 가능합니다.");
        form.id.select();
        return false;
    }
    else if(form.passwd.value.length < 5){
        alert("비밀번호는 5자 이상 입력 가능합니다.");
        form.passwd.focus();
        return false;
    }
    form.submit();
}
```

validation.js



유효성 검사(validation)

자바스크립트 charAt() 메서드

The charAt() method returns the character at the specified index in a string.

The index of the first character is 0, the second character is 1, and so on.

Tip: The index of the last character in a string is *string.length-1*, the second last character is *string.length-2*, and so on (See "More Examples").

```
<!DOCTYPE html>
<html>
<body>

<p>Click the button to display the first character of the string
"HELLO WORLD".</p>

<button onclick="myFunction()">Try it</button>

<p id="demo"></p>

<script>
function myFunction() {
  var str = "HELLO WORLD";
  var res = str.charAt(0)
  document.getElementById("demo").innerHTML = res;
}
</script>

</body>
</html>
```

Click th

Try it

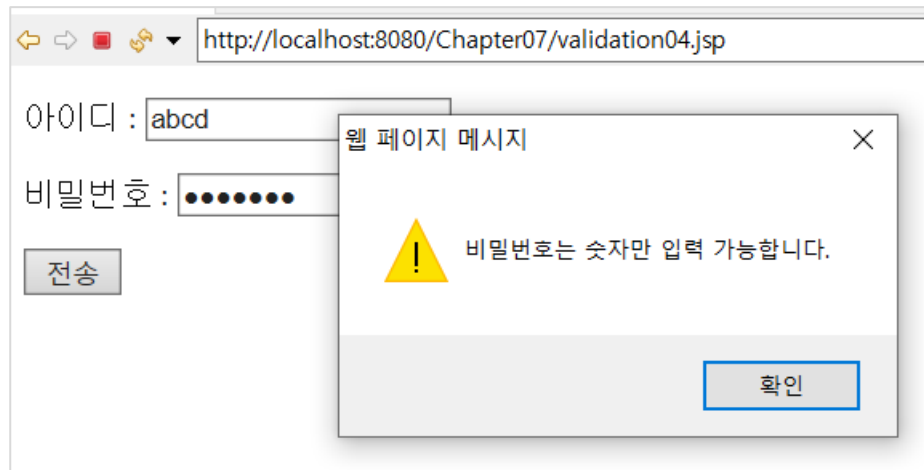
H



유효성 검사(validation)

폼 페이지에 입력한 데이터 값의 숫자 여부 확인하기

isNaN(document.폼 이름.입력양식 이름.value)



유효성 검사(validation)

폼 페이지에 입력한 데이터 값의 숫자 여부 확인하기

`isNaN(document.폼 이름.입력양식 이름.value)`

```
function checkLogin(){
    var form = document.loginForm;

    for(var i=0; i<form.userid.value.length; i++){
        var ch = form.userid.value.charAt(i);

        if((ch<'a' || ch>'z') && (ch>'A' || ch<'Z') && (ch>'0' || ch<'9')){
            alert("아이디는 영문 소문자만 가능합니다.");
            form.userid.select();
            return false;
        }

        if(isNaN(form.passwd.value)){
            alert("비밀번호는 숫자만 입력 가능합니다.");
            form.passwd.select();
            return false;
        }
    }

    form.submit();
}
```

checklogin.js



유효성 검사(validation)

폼(form)에 onsubmit 속성 사용하기

```
<title>onsubmit 예제</title>
<script>
    function checkForm(){
        var form = document.regForm;
        var pwd = form.passwd.value;

        if(pwd==""){
            alert("비밀번호를 입력해주세요");
            form.passwd.focus();
            return false;    //빈 문자열 반환
        }
    }

```

```
</script>
</head>
<body>
```

```
    <form name="regForm" action="add_process.jsp" method="post"
        onsubmit="return checkForm()">
        <p> 아이디 : <input type="text" name="id">
        <p> 비밀번호 : <input type="text" name="passwd">
        <p> <input type="submit" value="전송">
    </form>
</body>
```

※ return 을 생략하면 action 페이지로 이동함

← → ↻ ⓘ localhost:8080/JSEExample/onsubmit.jsp ☆ EX ⚙ ⛔

아이디 : admin

비밀번호 :

전송

localhost:8080 내용:
비밀번호를 입력해주세요

확인



정규 표현식(Regular Expression)

정규 표현식(Regular Expression)

- 특정한 규칙을 가진 문자열의 집합을 표현하고 처리하는 것을 말한다.
- 데이터의 유효성 검사에 주로 사용된다.
- 자주 사용하는 정규 표현식

표현식	설 명
<code>^[0-9]*\$</code>	숫자
<code>^[a-zA-Z]*\$</code>	영문 대, 소문자
<code>^[가-힣]*\$</code>	한글
<code>^01(?:0 1 [6-9])-(?:\d{3} \d{4})-\d{4}\$</code>	휴대폰
<code>\d{6}-[1-4]{6}\$</code>	주민등록번호
<code>\d{3}-\d{2}\$</code>	우편번호

메타문자	설 명
<code>^</code>	정규식 시작
<code>*</code>	문자 반복
<code>\$</code>	정규식 끝
<code>[x]</code>	x를 찾음
<code>\d</code>	숫자(decimal)
<code>\s</code>	공백(space)
<code>\w</code>	알파벳+숫자(word)
<code>{3}</code>	반복횟수

정규 표현식(Regular Expression)

정규 표현식(Regular Expression) - Java

Package `java.util.regex`

Class `Pattern`

`java.lang.Object`
`java.util.regex.Pattern`

All Implemented Interfaces:

`Serializable`

```
public final class Pattern
extends Object
implements Serializable
```

A compiled representation of a regular expression.

A regular expression, specified as a string, must first be compiled into a `Pattern` object. Once compiled, the `Pattern` object can be used to create `Matcher` objects that can match arbitrary character sequences against the regular expression. Multiple `Matcher` objects can share the same pattern.

A typical invocation sequence is thus

```
Pattern p = Pattern.compile("a*b");
Matcher m = p.matcher("aaaaab");
boolean b = m.matches();
```



정규 표현식(Regular Expression)

정규 표현식(Regular Expression) - Java

```
package regex;
import java.util.regex.Matcher;

public class RegExTest {
    public static void main(String[] args) {
        Pattern p = Pattern.compile("a*b");
        Matcher m = p.matcher("aaaaab");
        boolean b1 = m.matches();
        System.out.println(b1);

        System.out.println("-----");

        String pattern = "[0-9]*"; //숫자만
        String value = "abc1031"; //매칭 문자열

        boolean b2 = Pattern.matches(pattern, value);
        System.out.println(b2);
    }
}
```

true

false



정규 표현식(Regular Expression)

정규 표현식(Regular Expression) - Java

```
String name = "세종대왕";  
String tel = "010-3355-7979";  
  
//유효성 검사  
boolean name_check = Pattern.matches("[가-힣]*$", name);  
boolean tel_check = Pattern.matches("^01(?:0|1|[6-9])-(?:\\d{3}|\\d{4})-\\d{4}$", tel);  
  
//출력  
System.out.println("이름 : " + name_check);  
System.out.println("전화번호 : " + tel_check);
```

```
이름 : true  
전화번호 : true
```



정규 표현식(Regular Expression)

정규 표현식(Regular Expression) - JavaScript

https://www.w3schools.com/js/js_regexp.asp

Regular Expression Modifiers

Modifiers can be used to perform case-insensitive more global searches:

Modifier	Description	Try it
i	Perform case-insensitive matching	Try it »
g	Perform a global match (find all matches rather than stopping after the first match)	Try it »
m	Perform multiline matching	Try it »

Regular Expression Patterns

Brackets are used to find a range of characters:

Expression	Description	Try it
[abc]	Find any of the characters between the brackets	Try it »
[0-9]	Find any of the digits between the brackets	Try it »
(x y)	Find any of the alternatives separated with	Try it »



정규 표현식(Regular Expression)

JavaScript String match() Method

[< Previous](#)[JavaScript String Reference](#)

match()는 정확히 일치하는 문자열을 반환(자바의 equals()와 같다)

Example

Search a string for "ain":

```
var str = "The rain in Spain";  
var res = str.match(/ain/);
```

[Try it Yourself »](#)

```
<p>Click the button to perform a global search for the letters "ain" in  
a string, and display the matches.</p>
```

```
<button onclick="myFunction()">Try it</button>
```

```
<p id="demo"></p>
```

```
<script>
```

```
function myFunction() {
```

```
    var str = "The rain in SPAIN stays mainly in the plain";
```

```
    var res = str.match(/ain/i);
```

```
    document.getElementById("demo").innerHTML = res;
```

```
}
```

```
</script>
```

```
</body>
```

```
</html>
```

[Try it](#)

ain



정규 표현식(Regular Expression)

JavaScript String search() Method

search()는 일치하는 문자열의 위치 (인덱스)를 반환함

```
<p>Click the button to search a string for "W3Schools", and display the position of the match.</p>
```

```
<button onclick="myFunction()">Try it</button>
```

```
<p id="demo"></p>
```

```
<script>
```

```
function myFunction() {
```

```
    var str = "Visit W3Schools!";
```

```
    var n = str.search("W3Schools");
```

```
    document.getElementById("demo").innerHTML = n;
```

```
}
```

```
</script>
```

정규 표현식(Regular Expression)

JavaScript test() Method

The test() method tests for a match in a string. —→ **test()는 문자열의 일치하는지 여부를 true/false로 반환.**

This method returns true if it finds a match, otherwise it returns false.

```
<!DOCTYPE html>
<html>
<body>

<p>The test() method returns true if it finds a match, otherwise it
returns false.</p>
<p>Click the button to search a string for the character "e".</p>

<button onclick="myFunction()">Try it</button>

<p id="demo"></p>

<script>
function myFunction() {
  var str = "The best things in life are free";
  var patt = new RegExp("e");
  var res = patt.test(str);
  document.getElementById("demo").innerHTML = res;
}
</script>

</body>
</html>
```

The test

Click the

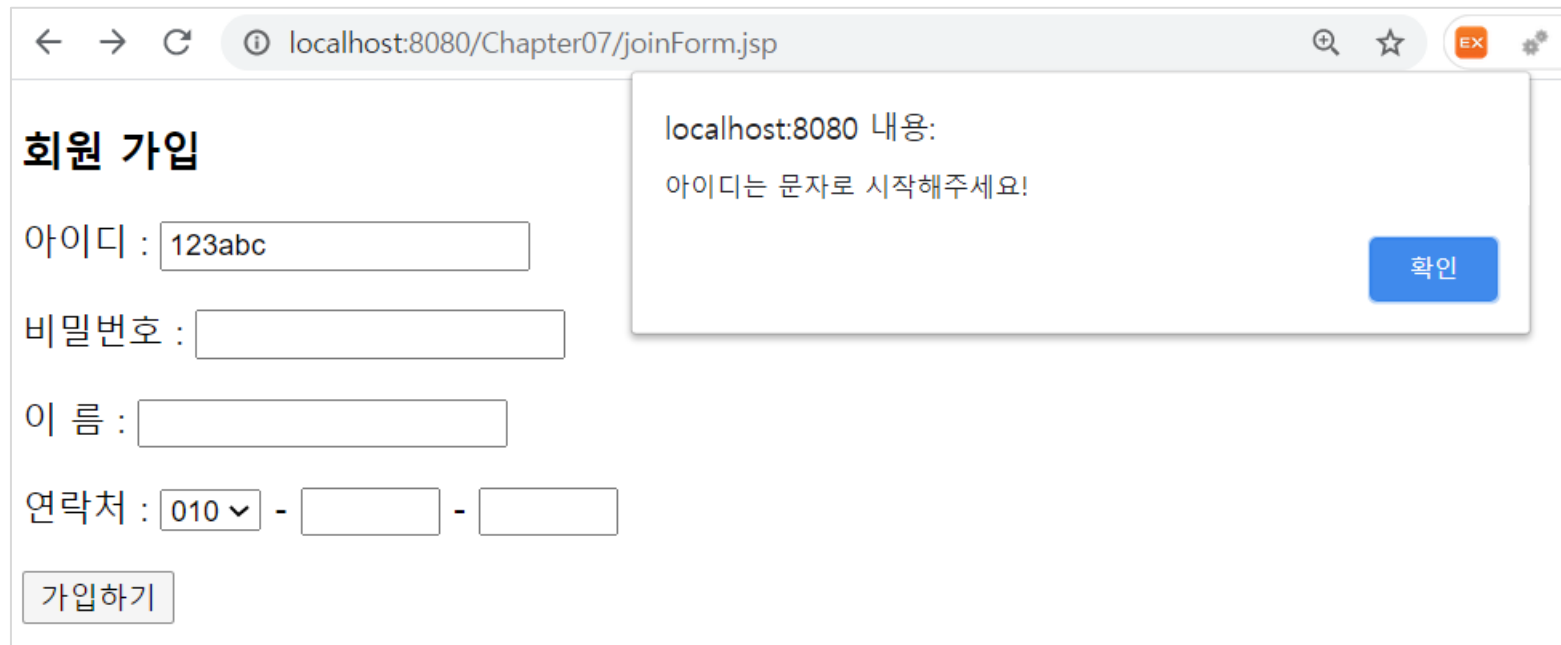
Try it

true



유효성 검사(validation)

회원 가입 폼에 입력한 데이터 형식 유효성 검사 – 정규 표현식



The screenshot shows a web browser window with the address bar displaying "localhost:8080/Chapter07/joinForm.jsp". The page title is "회원 가입" (Member Registration). The form contains the following fields:

- 아이디 (ID): 123abc
- 비밀번호 (Password): [Empty]
- 이름 (Name): [Empty]
- 연락처 (Contact): 010 - [Empty] - [Empty]

At the bottom of the form is a button labeled "가입하기" (Sign Up). A validation error message is displayed in a white box with a blue "확인" (Confirm) button:

localhost:8080 내용:
아이디는 문자로 시작해주세요!

유효성 검사(validation)

회원 가입 폼에 입력한 데이터 형식 유효성 검사 – 정규 표현식

```
<form name="Member" action="joinProcess.jsp" method="post">
  <p>
    <label for="id">아이디 : </label>
    <input type="text" id="id" name="userid">
  </p>
  <p>
    <label for="passwd">비밀번호 : </label>
    <input type="password" id="passwd" name="passwd">
  </p>
  <p>
    <label for="name">이름 : </label>
    <input type="text" id="name" name="name">
  </p>
  <p>
    <label for="phone">연락처 : </label>
    <select name="phone1">
      <option value="010">010</option>
      <option value="011">011</option>
      <option value="016">016</option>
      <option value="017">017</option>
      <option value="019">019</option>
    </select>
    - <input type="text" maxlength="4" size="4" name="phone2">
    - <input type="text" maxlength="4" size="4" name="phone3">
  </p>
  <p><input type="button" value="가입하기" onclick="checkMember()"></p>
</form>
```



유효성 검사(validation)

checkmember.js

```
function checkMember(){
    var regExId = /^[a-zA-Zㄱ-ㅎ|ㅏ-ㅣ|가-힣]/; //문자로 시작
    var regExPwd = /^[0-9]*$/; //숫자만
    var regExName = /^[가-힣]*$/; //한글만
    var regExPhone = /^\\d{3}-\\d{3,4}-\\d{4}$/; //연락처 자리수

    var form = document.Member;

    var id = form.userid.value;
    var passwd = form.passwd.value;
    var name = form.name.value;
    var phone = form.phone1.value + "-" + form.phone2.value
                + "-" + form.phone3.value;
```

유효성 검사(validation)

```
if(!regExId.test(id)){  
    alert("아이디는 문자로 시작해주세요!");  
    form.id.select();  
    return false;  
}  
  
if(!regExPwd.test(passwd)){  
    alert("비밀번호는 숫자만 입력해주세요!");  
    return false;  
}  
  
if(!regExName.test(name)){  
    alert("이름은 한글만 입력해주세요");  
    return false;  
}  
  
if(!regExPhone.test(phone)){  
    alert("연락처 입력을 확인해 주세요!");  
    return false;  
}  
form.submit();
```



유효성 검사(validation)

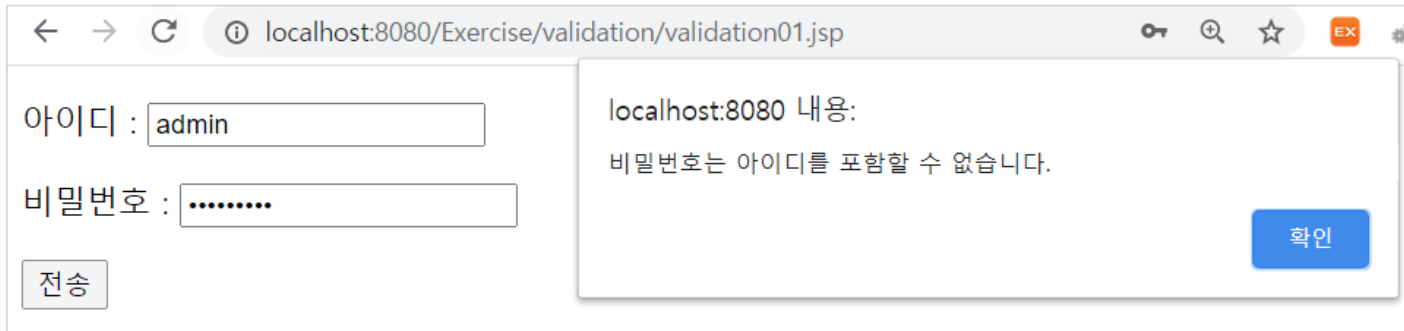
join_process.jsp

```
<%  
    request.setCharacterEncoding("utf-8");  
%>
```

```
<h3>등록 내용</h3>  
<p>아이디 : <%=request.getParameter("userid") %>  
<p>비밀번호 : <%=request.getParameter("passwd") %>  
<p>이름 : <%=request.getParameter("name") %>  
<p>연락처 : <%=request.getParameter("phone1") %>  
            - <%=request.getParameter("phone2") %>  
            - <%=request.getParameter("phone3") %>
```


Session 연습문제

■ 실습 예제



The screenshot shows a web browser window with the address bar displaying `localhost:8080/Exercise/validation/validation01.jsp`. The page contains a login form with two input fields: "아이디 : admin" and "비밀번호 :". Below the password field is a "전송" (Submit) button. An error message box is displayed over the form, titled "localhost:8080 내용:", with the text "비밀번호는 아이디를 포함할 수 없습니다." (Password cannot contain the ID). A blue "확인" (OK) button is located at the bottom right of the message box.

1. validation01.jsp 파일 생성

- 1) 아이디, 비밀번호가 입력되지 않으면 "아이디(또는 비밀번호)가 입력을 입력해주세요"라는 메시지 창이 뜨도록 작성한다.
- 2) 비밀번호에 아이디 값을 포함하면 "비밀번호는 아이디를 포함할 수 없습니다."라는 메시지 창이 뜨도록 작성한다.

2. validation_process.jsp 파일 생성하여 요청 파라미터값을 출력한다.



Session 연습문제

■ 실습 예제

```
function checkForm(){
    var form = document.loginForm;
    var id = form.userid.value;
    var pwd = form.passwd.value;

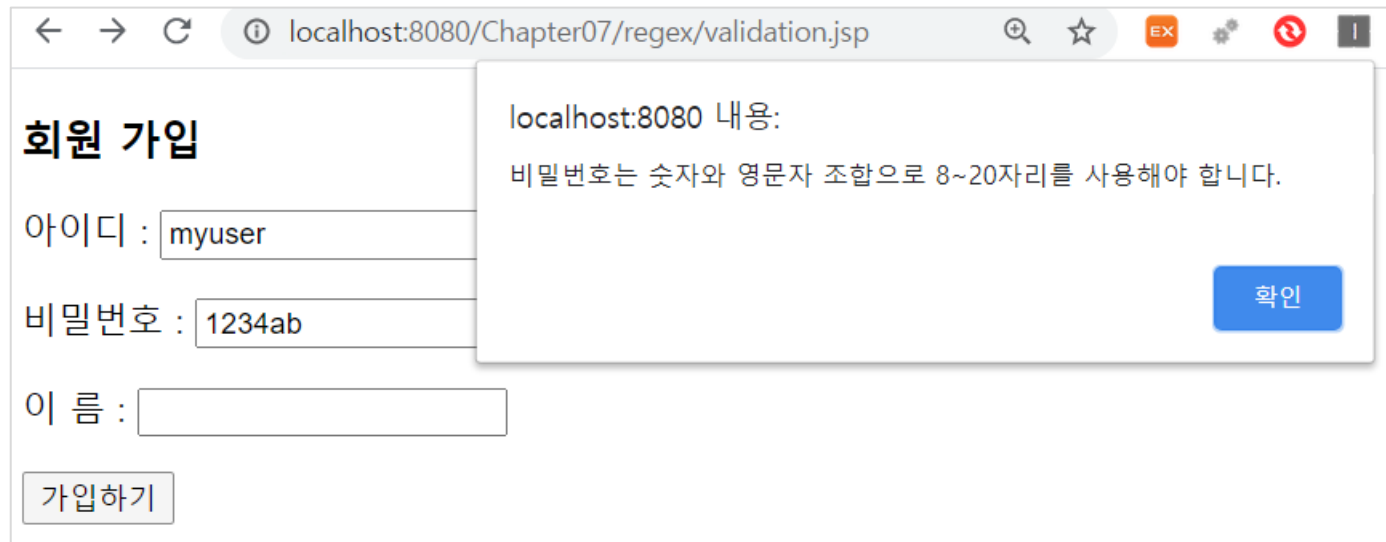
    if(id.length == ""){
        alert("아이디를 입력해주세요.");
        form.userid.focus();
        return false;
    }
    else if(pwd==""){
        alert("비밀번호를 입력해주세요.");
        form.pwd.focus();
        return false;
    }
    if(pwd.search(id) > -1){
        alert("비밀번호는 아이디를 포함할 수 없습니다.");
        form.pwd.select();
        return false;
    }
    form.submit();
}
```

pwd.match(id)도 가능함



유효성 검사(validation)

비밀번호 숫자와 영문자의 조합으로 입력하기



The screenshot shows a web browser window with the address bar displaying `localhost:8080/Chapter07/regex/validation.jsp`. The page content includes a form titled "회원 가입" (Member Registration) with the following fields:

- 아이디 (ID):
- 비밀번호 (Password):
- 이름 (Name):

Below the fields is a button labeled "가입하기" (Join). A modal dialog box is open over the password field, displaying the message:

localhost:8080 내용:
비밀번호는 숫자와 영문자 조합으로 8~20자리를 사용해야 합니다.

The dialog box has a blue button labeled "확인" (Confirm).

유효성 검사(validation)

비밀번호 숫자와 영문자의 조합으로 입력하기

```
<h3>회원 가입</h3>
<form name="Member" action="validation_process.jsp" method="post">
  <p>
    <label for="id">아이디 : </label>
    <input type="text" id="id" name="userid">
  </p>
  <p>
    <label for="passwd">비밀번호 : </label>
    <input type="text" id="passwd" name="passwd">
  </p>
  <p>
    <label for="name">이름 : </label>
    <input type="text" id="name" name="name">
  </p>
  <p><input type="button" value="가입하기" onclick="checkMember()"></p>
</form>
```

유효성 검사(validation)

회원 가입 폼에 입력한 데이터 형식 유효성 검사 – 정규 표현식

```
function checkMember(){
    var form = document.Member;
    var pwd = form.passwd.value;

    var regExpPwd = /^[a-zA-Z0-9]{8,20}$/;
    var chk_num = pwd.search(/[0-9]/g);
    var chk_eng = pwd.search(/[a-zA-Z]/g);

    if(!regExpPwd.test(pwd)){
        alert('비밀번호는 숫자와 영문자 조합으로 8~20자리를 사용해야 합니다.');
```

return false;

```
    }

    if(chk_num < 0 || chk_eng < 0){
        alert('비밀번호는 숫자와 영문자를 혼용하여야 합니다.');
```

return false;

```
    }

    form.submit();
}
```

