# 4. 파일 입출력

## ❖ 파일로부터 데이터를 읽는 방법

- 스탠드얼론 프로그램에서와 마찬가지로 java.io.FileReader 또는 java.io.FileInputStream 클래스를 이용하면 된다.

**[예제3-13] 텍스트 파일을 읽는 스탠드얼론 프로그램**

```
import java.io.*;
class FileReadProgram {
    public static void main(String args[]) {
        BufferedReader reader = null;
        try {
            reader = new BufferedReader(new FileReader( "input.txt "));
            while (true) {
                String str = reader.readLine();
                if (str == null)
                    break;
                System.out.println(str);
            }
        }
        catch (FileNotFoundException fnfe) {
            System.out.println( "파일이 존재하지 않습니다. " );
        }
        catch (IOException ioe) {
            System.out.println( " 파일을 읽을 수 없습니다. " );
        }
        finally {
            try {
                reader.close();
            }
            catch (Exception e) {
            }
        }
    }
}
```

파일을 연다

한 행의 텍스트 데이터를 읽는다

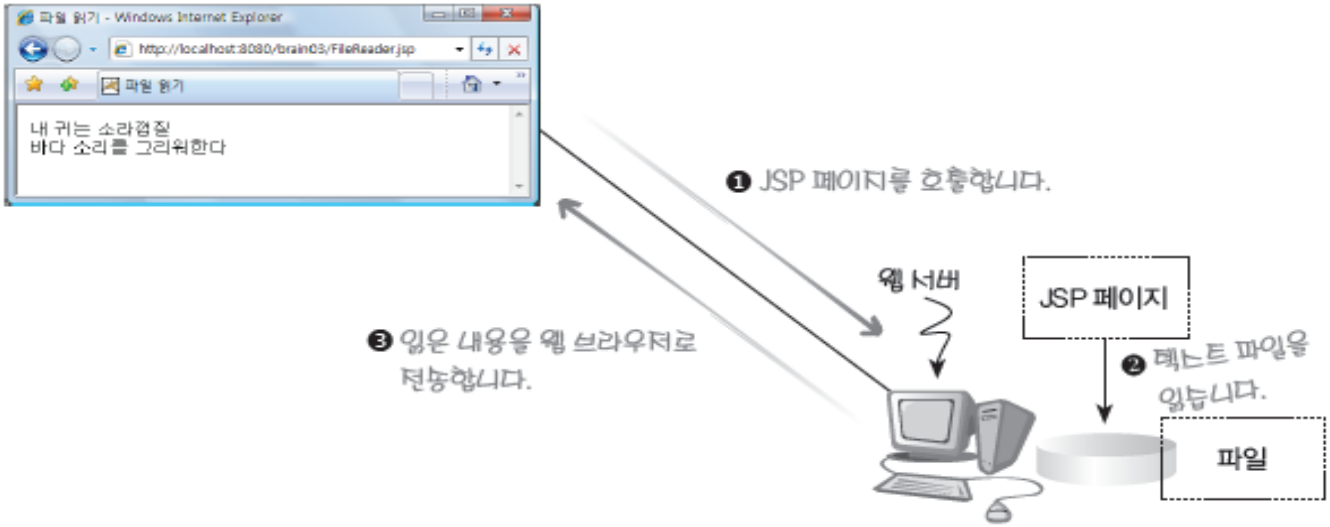
읽은 데이터를 모니터로 출력한다

파일을 닫는다

# 4. 파일 입출력

## ❖ 파일로부터 데이터를 읽는 방법

- 파일을 읽는 웹 애플리케이션의 구성도



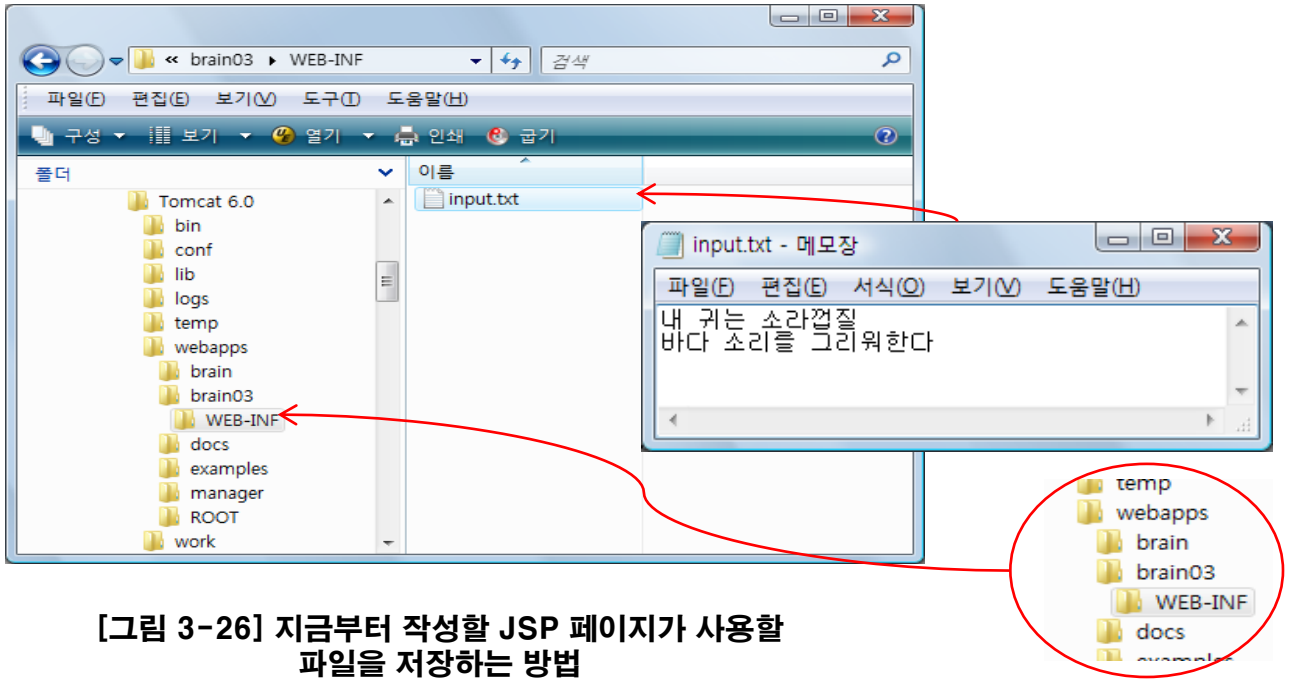

[그림 3-25] 텍스트 파일을 읽어서 출력하는 웹 애플리케이션의 구성도

- JSP 페이지나 서블릿 클래스는 스탠드얼론 프로그램과 달리 웹 서버의 일부가 되어서 실행된다.

# 4. 파일 입출력

## ❖ 파일로부터 데이터를 읽는 방법

- 웹 브라우저에서 해당 파일의 URL을 통해 파일의 내용을 직접 읽을 수 없도록 만들려면 WEB-INF 디렉터리에 저장해야 한다.




[그림 3-26] 지금부터 작성할 JSP 페이지가 사용할 파일을 저장하는 방법

# 4. 파일 입출력

## ❖ 파일로부터 데이터를 읽는 방법

- JSP 페이지 안에서 파일을 읽기 위해서는 기본적으로 파일의 절대 경로를 사용하거나 톰캣의 설치 디렉터리로부터 상대 경로명을 사용해야 한다.
- 하지만 더 좋은 방법은 getRealPath 메서드를 이용해서 웹 애플리케이션 내에서의 경로명을 절대 경로명으로 바꾸어 사용하는 방법이다.

```
String filePath = application.getRealPath( "/WEB-INF/input.txt ");
```

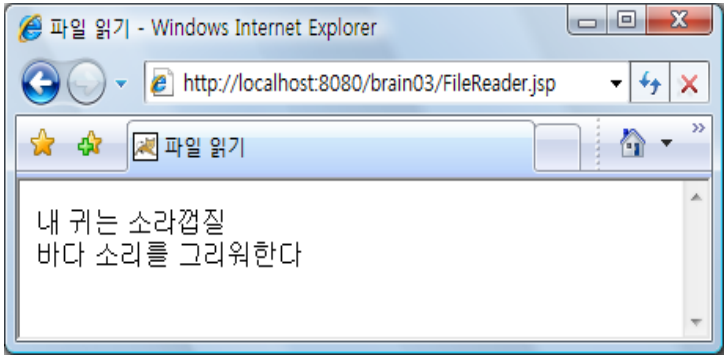
웹 애플리케이션 디렉터리 내에서의 파일의 경로명

# 4. 파일 입출력

## ❖ 파일로부터 데이터를 읽는 방법

**[예제3-14] 텍스트 파일의 내용을 읽어서 웹 브라우저로 출력하는 JSP 페이지**

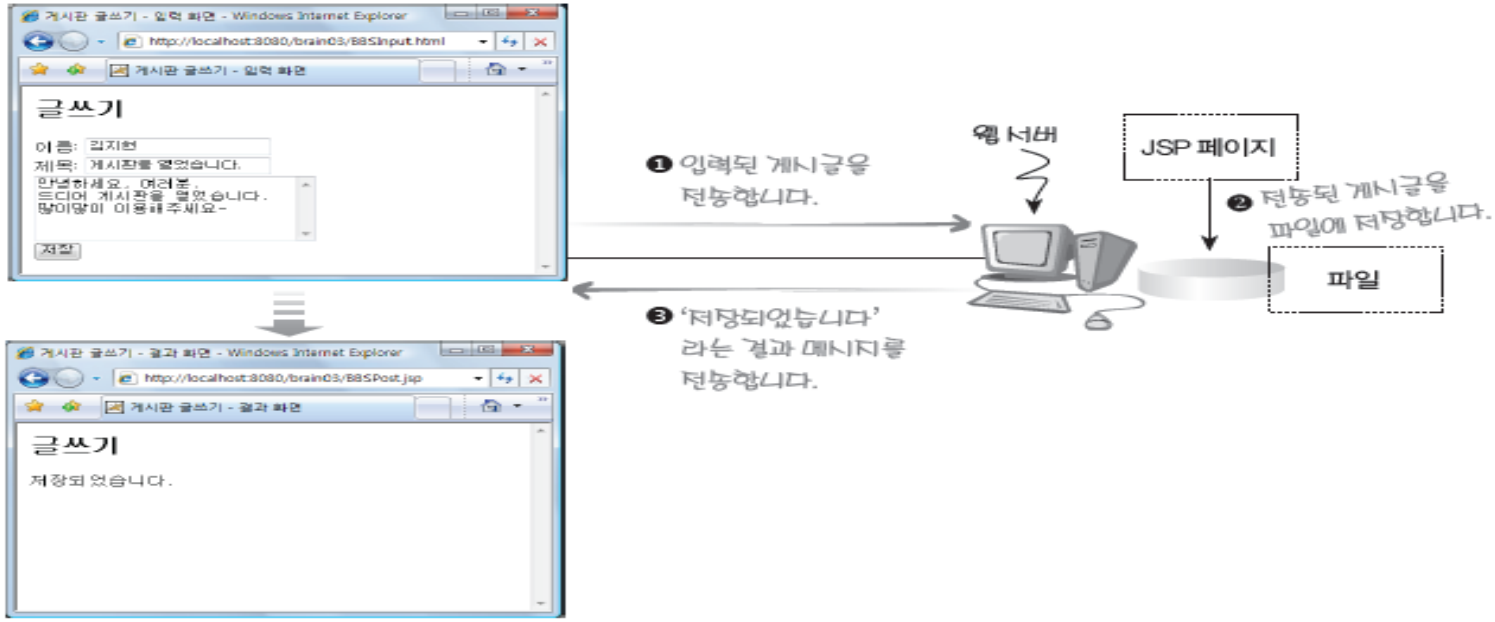
```
<%@page contentType= “text/html; charset=euc-kr ” %>
<%@page import= “java.io.* ” %>
<HTML>
  <HEAD><TITLE> 파일 읽기</TITLE></HEAD>
  <BODY>
    <%
      BufferedReader reader = null;
      try {
        String filePath = application.getRealPath( “/WEB-INF/input.txt ”);
        reader = new BufferedReader(new FileReader(filePath));
        while (true) {
          String str = reader.readLine();
          if (str == null)
            break;
          out.println(str + “<BR> ”);
        }
      }
      catch (FileNotFoundException fnfe) {
        out.println( “파일이 존재하지 않습니다. ” );
      }
      catch (IOException ioe) {
        out.println( “파일을 읽을 수 없습니다. ” );
      }
      finally {
        try {
          reader.close();
        }
        catch (Exception e) {
        }
      }
    %>
  </BODY>
</HTML>
```



# 4. 파일 입출력

## ❖ 파일에 데이터를 쓰는 방법

- JSP 페이지에서도 스탠드얼론 프로그램에서와 마찬가지로 java.io.FileWriter, java.io.PrintWriter, java.io.FileOutputStream 등의 클래스를 이용하면 된다.




[그림 3-28] 게시판 글쓰기 기능을 구현하는 웹 애플리케이션의 구성도

# 4. 파일 입출력

## ❖ 파일에 데이터를 쓰는 방법

- 앞 페이지 두 화면의 URL을 정하고, URL에 해당하는 HTML 문서를 작성한다.

**http://localhost:8080/brain03/BBSInput.html** ← (그림3-29) 위쪽 화면 URL

**http://localhost:8080/brain03/BBSPost.jsp** ← (그림3-29) 아래쪽 화면 URL

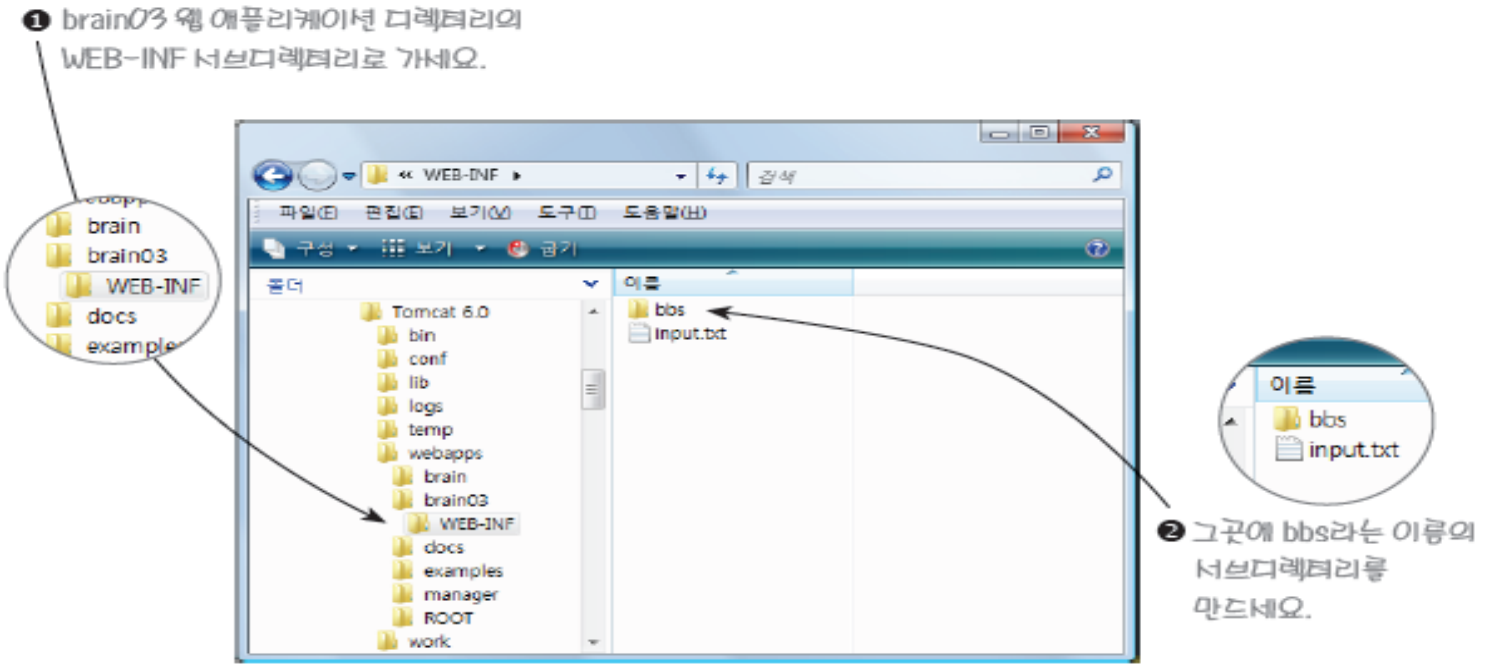
**[예제3-15] 게시판 글쓰기 기능의 입력 화면을 구현하는 HTML 문서**

```
<HTML>
    <HEAD>
        <META http-equiv= “Content-Type” content= “text/html;charset=euc-kr ”>
        <TITLE>게시판 글쓰기 - 입력 화면</TITLE>
    </HEAD>
    <BODY>
        <H2>글쓰기</H2>
        <FORM ACTION=BBSPost.jsp METHOD=POST>
            이름: <INPUT TYPE=TEXT NAME=NAME><BR>
            제목: <INPUT TYPE=TEXT NAME=TITLE><BR>
            <TEXTAREA COLS=30 ROWS=5 NAME=CONTENT></TEXTAREA><BR>
            <INPUT TYPE=SUBMIT VALUE= ‘저장 ’  >
        </FORM>
    </BODY>
</HTML>
```

# 4. 파일 입출력

## ❖ 파일에 데이터를 쓰는 방법

- 이 예제는 HTML 문서를 통해 게시글을 입력받을 때마다 그 내용을 담은 파일을 새로 하나씩 만들어야 한다.




[그림 3-29] 게시글 파일을 저장할 디렉터리 만들기

# 4. 파일 입출력

## ❖ 파일에 데이터를 쓰는 방법

- 밀리세컨드 단위의 현재 시각을 파일 이름으로 사용하기로 한다. 이 값은 java.util.Date 클래스의 객체를 만든 다음 getTime 메서드를 호출해서 얻을 수 있다.

```
Date date = new Date();

long time = date.getTime();
```

현재 시각을 밀리세컨드 단위로 가져오는 메서드

- getTime 메서드가 리턴하는 값은 1212414054907처럼 아주 긴 정수이다.

# 4. 파일 입출력

## ❖ 파일에 데이터를 쓰는 방법

**[예제3-16] 게시글을 저장하는 JSP 페이지**

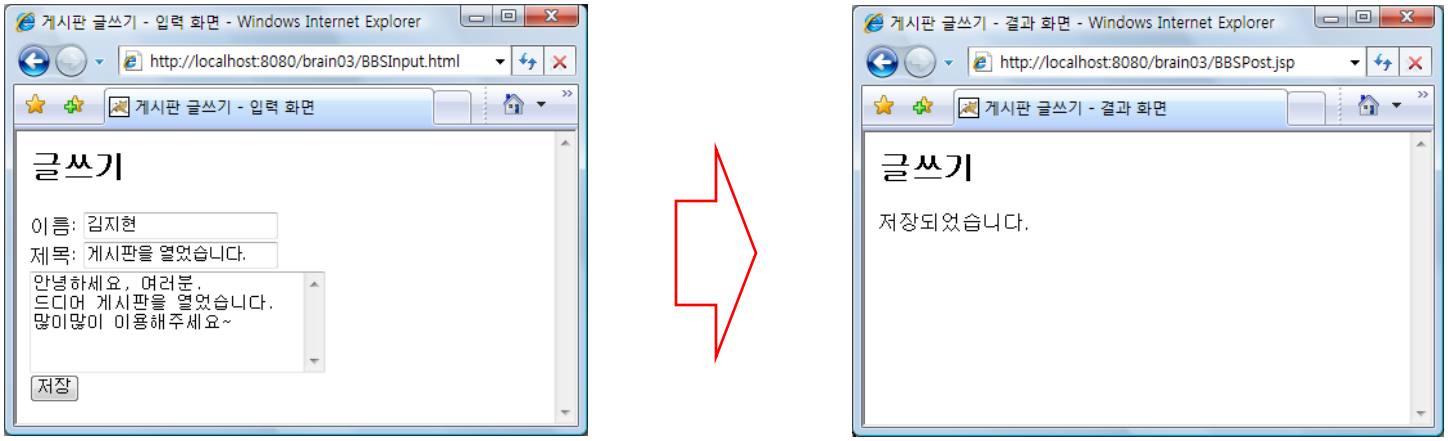
```
<%@page contentType= "text/html; charset=euc-kr "%>
<%@page import= "java.io.*, java.util.Date "%>
<HTML>
    <HEAD><TITLE>게시판 글쓰기 - 결과 화면</TITLE></HEAD>
    <BODY>
        <H2>글쓰기</H2>
        <%
            request.setCharacterEncoding( "euc-kr ");
            String name = request.getParameter( "NAME ");
            String title = request.getParameter( "TITLE ");
            String content = request.getParameter( "CONTENT ");
            Date date = new Date();
            Long time = date.getTime();
            String filename = time + ".txt ";
            PrintWriter writer = null;
            try {
                String filePath = application.getRealPath(
                                        "/WEB-INF/bbs/ " + filename);
                writer.printf( "제목: %s %n ", title);
                writer.printf( "글쓴이: %s %n ", name);
                writer.println(content);
                out.println( "저장되었습니다. " );
            }
            catch (IOException ioe) {
```

**[예제3-16] 게시글을 저장하는 JSP 페이지**

```
                out.println( "파일에 데이터를 쓸 수 없습니다. " );
            }
            finally {
                try {
                    writer.close();
                }
                catch (Exception e) {
                }
            }
        %>
    </BODY>
</HTML>
```

# 4. 파일 입출력

## ❖ 파일에 데이터를 쓰는 방법




[그림 3-30] 예제 3-15, 3-16의 실행 방법

- 이 애플리케이션의 문제점 : 결과 화면에서 새로 고침 버튼을 누를 때마다 게시글 디렉터리에 똑같은 내용의 게시글 파일이 하나씩 더 생긴다.
  - 웹 브라우저의 새로 고침 버튼을 누를 때 마다 [예제 3-16]의 JSP 페이지가 다시 호출되기 때문이다.

# 4. 파일 입출력

## ❖ 파일에 데이터를 쓰는 방법

[예제 3-17] 게시글을 저장하는 JSP 페이지와 그 결과를 출력하는 JSP 페이지

게시글을 파일에 저장하는 JSP 페이지

```
01  <%@page contentType="text/html; charset=euc-kr"%>
02  <%@page import="java.io.*, java.util.Date"%>
03  <%
04      request.setCharacterEncoding("euc-kr");
05      String name = request.getParameter("NAME");
06      String title = request.getParameter("TITLE");
07      String content = request.getParameter("CONTENT");
08      Date date = new Date();
09      Long time = date.getTime();
10      String filename = time + ".txt";
11      String result;
12      PrintWriter writer = null;
13      try {
14          String filePath =
                    application.getRealPath("/WEB-INF/bbs/" + filename);
15          writer = new PrintWriter(filePath);
16          writer.printf("제목: %s %n", title);
17          writer.printf("글쓴이: %s %n", name);
18          writer.println(content);
19          result = "SUCCESS";
20      }
21      catch (IOException ioe) {
22          result = "FAIL";
23      }
24      finally {
25          try {
26              writer.close();
27          }
28          catch (Exception e) {
29          }
30      }
31      response.sendRedirect("BBSPostResult.jsp?RESULT=" + result);
32  %>
```

실행의 결과를 URL 뒤에 붙여서 전송합니다.

# 4. 파일 입출력

## ❖ 파일에 데이터를 쓰는 방법

파일 저장 결과를 출력하는 JSP 페이지

```
01  <%@page contentType="text/html; charset=euc-kr"%>
02  <HTML>
03      <HEAD><TITLE>게시판 글쓰기 - 결과 화면</TITLE></HEAD>
04      <BODY>
05          <H2>글쓰기</H2>
06          <%
07              String str = request.getParameter("RESULT");
08              if (str.equals("SUCCESS"))
09                  out.println("저장되었습니다.");
10              else
11                  out.println("파일에 데이터를 쓸 수 없습니다.");
12          %>
13      </BODY>
14  </HTML>
```

실행의 결과를 가져옵니다.

# 5. 다른 JSP 페이지 호출하기

## ❖ forward 메서드의 사용 방법

- forward 메서드는 JSP 페이지 안에서 다른 JSP 페이지를 호출할 때 사용하는 메서드이다.
- 이 메서드는 javax.servlet.RequestDispatcher 인터페이스에 속하기 때문에 이 타입의 객체가 있어야 호출할 수 있다.

```
RequestDispatcher dispatcher = request.getRequestDispatcher( "Result.jsp ");
```

호출할 JSP 페이지의 URL 경로명

- forward 메서드를 호출할 때에는 request 내장 변수와 response 내장 변수를 파라미터로 넘겨줘야 한다.

```
dispatcher.forward(request, response);
```

request 내장 변수 response 내장 변수

# 5. 다른 JSP 페이지 호출하기

## ❖ forward 메서드의 사용 방법

- forward 메서드를 통해 호출하는 JSP 페이지로 데이터를 넘겨주려면 이 메서드보다 먼저 request.setAttribute 메서드를 호출해서 request 내장 변수 안에 데이터를 저장해 놓아야 한다.

```
request.setAttribute( “HEIGHT ”, new Integer(178));
```

데이터 이름 데이터 값

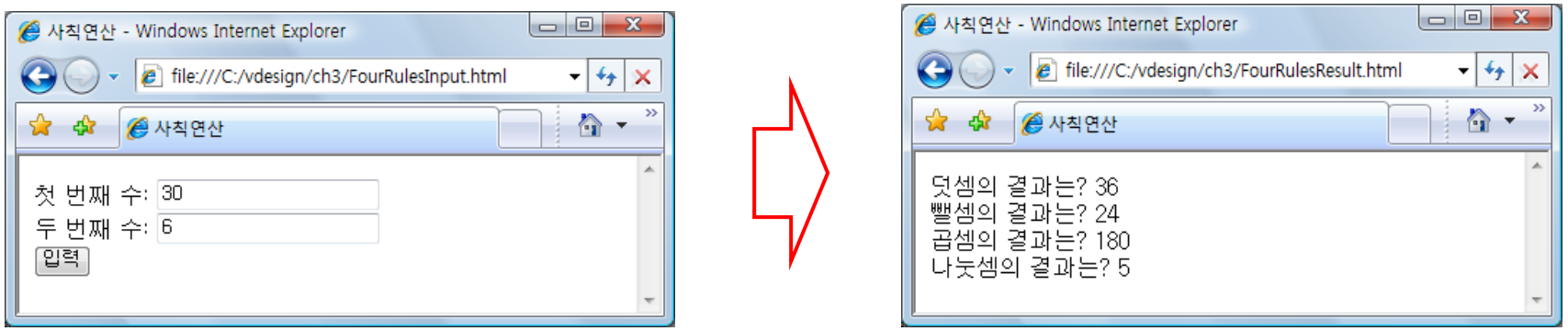
- 호출된 JSP 페이지 안에서 request 내장 변수 안의 데이터를 가져오려면 request.getAttribute 메서드를 호출하면 된다.

```
Integer height = (Integer) request.getAttribute( “HEIGHT ”);
```

캐스트 연산자 데이터 이름

# 5. 다른 JSP 페이지 호출하기

## ❖ forward 메서드의 사용 방법




[그림 3-32] 사칙 연산을 수행하는 웹 애플리케이션 화면 설계

(그림 3-32)의 왼쪽 화면의 URL

**http://localhost:8080/brain03/FourRules.html**

(그림 3-32)의 오른쪽 화면의 URL

**http://localhost:8080/brain03/FourRules.jsp**

사칙 연산을 수행하는 JSP 페이지의 URL

**http://localhost:8080/brain03/FourRuelsResult.jsp**

사칙 연산의 결과를 출력하는 JSP 페이지의 URL

# 5. 다른 JSP 페이지 호출하기

## ❖ forward 메서드의 사용 방법

**[예제3-18] 두 개의 수를 입력받는 HTML 문서**

```
<HTML>
    <HEAD>
        <META http-equiv= “Content-Type ” content= “text/html;charset=euc-kr ”>
        <TITLE>사칙 연산</TITLE>
    </HEAD>
    <BODY>
        <FORM ACTION=FourRules.jsp>
            첫 번째 수: <INPUT TYPE=TEXT NAME=NUM1><BR>
            두 번째 수: <INPUT TYPE=TEXT NAME=NUM2><BR>
            <INPUT TYPE=SUBMIT VALUE= ‘입력 ’  >
        </FORM>
    </BODY>
</HTML>
```

# 5. 다른 JSP 페이지 호출하기

## ❖ forward 메서드의 사용 방법

[예제 3-19] 사칙 연산을 수행하는 JSP 페이지와 그 결과를 출력하는 JSP 페이지

사칙 연산을 수행하는 JSP 페이지

```
01  <%
02      String str1 = request.getParameter("NUM1");
03      String str2 = request.getParameter("NUM2");
04      int num1 = Integer.parseInt(str1);
05      int num2 = Integer.parseInt(str2);
06      request.setAttribute("SUM", new Integer(num1 + num2));
07      request.setAttribute("DIFFERENCE", new Integer(num1 - num2));
08      request.setAttribute("PRODUCT", new Integer(num1 * num2));
09      request.setAttribute("QUOTIENT", new Integer(num1 / num2));
10      RequestDispatcher dispatcher =
                request.getRequestDispatcher("FourRulesResult.jsp");
11      dispatcher.forward(request, response);
12  %>
```

사칙 연산의 결과를 request 내장 변수 안에 저장합니다.

아래쪽 JSP 페이지를 호출합니다.

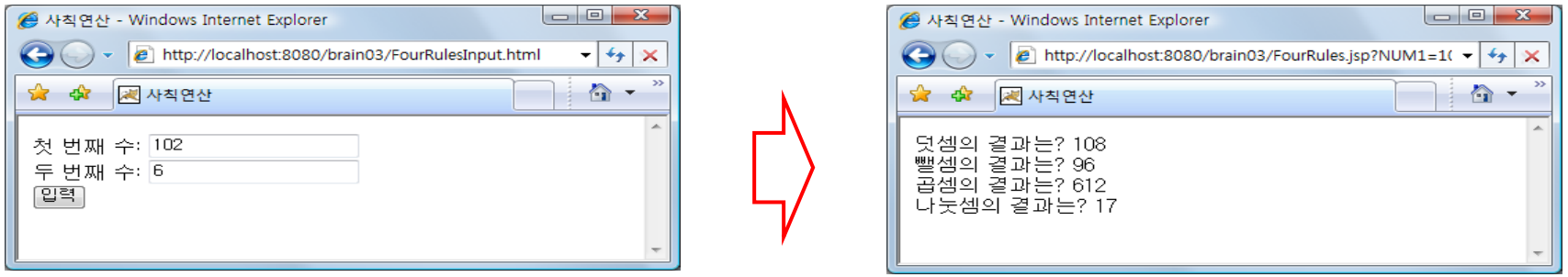
사칙 연산 결과를 출력하는 JSP 페이지

```
01  <%@page contentType="text/html; charset=euc-kr"%>
02  <HTML>
03      <HEAD><TITLE>사칙 연산</TITLE></HEAD>
04      <BODY>
05          덧셈의 결과는? <%= request.getAttribute("SUM") %> <BR>
06          뺄셈의 결과는? <%= request.getAttribute("DIFFERENCE") %> <BR>
07          곱셈의 결과는? <%= request.getAttribute("PRODUCT") %> <BR>
08          나눗셈의 결과는? <%= request.getAttribute("QUOTIENT") %> <BR>
09      </BODY>
10  </HTML>
```

request 내장 변수 안에 있는 데이터를 가져와서 출력합니다.

# 5. 다른 JSP 페이지 호출하기

## ❖ forward 메서드의 사용 방법

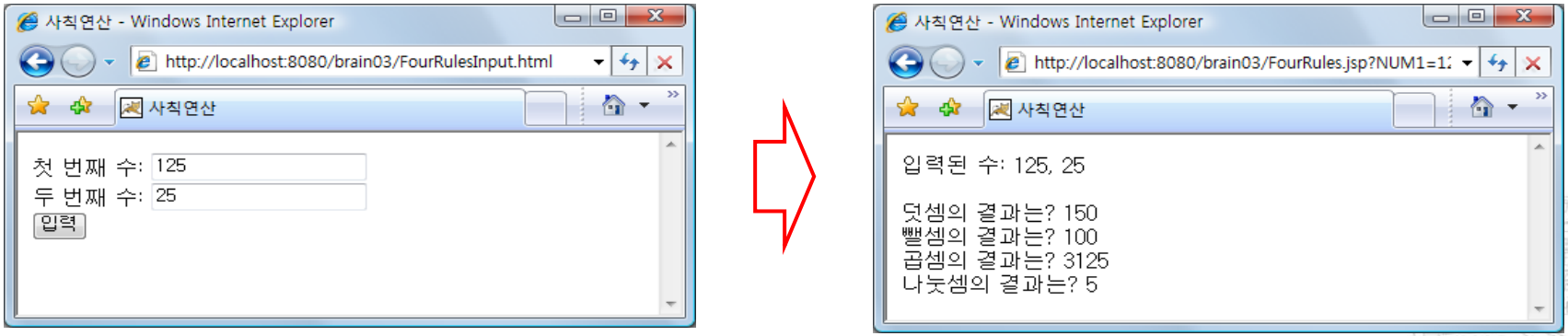



[그림 3-33] 예제 3-18, 3-19의 실행 결과

- forward 메서드가 request 내장 변수를 통해 전달하는 것은 애트리뷰트뿐만이 아니다. request 내장 변수 안에 있는 모든 데이터가 함께 전달된다.

```
<BODY>
    입력된 수: <%= request.getParameter( "NUM1 ") %>,
               <%= request.getParameter( "NUM2 ") %> <BR><BR>
    덧셈의 결과는? <%= request.getAttribute( "SUM ") %> <BR>
```

(예제 3-19)의 두 번째 JSP 페이지에 이 두 행을 추가한다



# 5. 다른 JSP 페이지 호출하기

## ❖ include 메서드의 사용 방법

- include 메서드도 forward 메서드처럼 다른 JSP 페이지를 호출하지만, 호출된 JSP 페이지가 끝나고 나면 실행 흐름의 제어가 본래의 JSP 페이지로 돌아온다.
- include 메서드는 forward 메서드와 마찬가지로 javax.servlet.RequestDispatcher 인터페이스에 속하므로 먼저 RequestDispatcher 객체를 구해야 한다.

```
RequestDispatcher dispatcher = request.getRequestDispatcher( "Today.jsp ");
```

호출할 JSP 페이지의 URL 경로명

- include 메서드를 호출할 때는 request 내장 변수와 response 내장 변수를 파라미터로 넘겨줘야 한다.

```
dispatcher.include(request, response);
```

request 내장 변수 response 내장 변수

# 5. 다른 JSP 페이지 호출하기

## ❖ include 메서드의 사용 방법

- include 메서드를 통해 호출되는 JSP 페이지로 데이터를 넘겨주기 위해서는 forward 메서드의 경우와 마찬가지로 request 내장 변수에 대해 serAttribute 메서드와 getAttribute 메서드를 호출하면 된다.

```
request.setAttribute( “WEIGHT ”, new Double(72.5));
```

request 내장 변수에
데이터를 저장하는 메서드

```
Double weight = (Double) request.getAttribute( “WEIGHT ”);
```

request 내장 변수에 저장되어
있는 데이터를 가져오는 메서드

# 5. 다른 JSP 페이지 호출하기

## ❖ include 메서드의 사용 방법

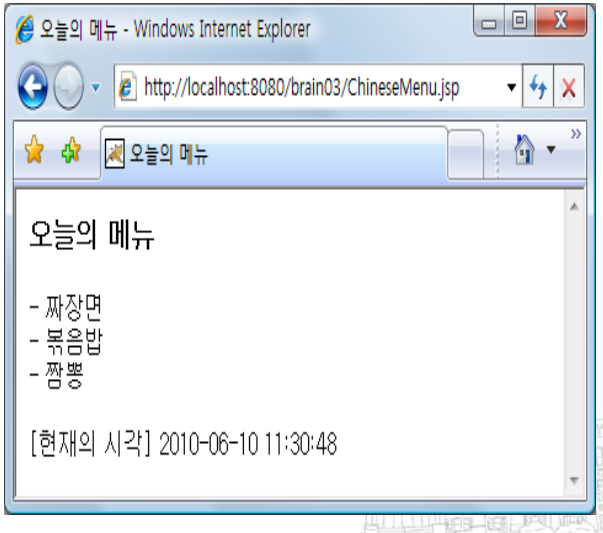
- include 메서드의 사용 예를 보여주는 JSP 페이지

**[예제3-20] include 메서드의 사용 예**

```
<%@page contentType= “text/html; charset=euc-kr ”%>
<HTML>
    <HEAD><TITLE>오늘의 메뉴</TITLE></HEAD>
    <BODY>
        <H3>오늘의 메뉴</H3>
        - 짜장면 <BR>
        - 볶음밥 <BR>
        - 짬뽕 <BR><BR>
        <%
            out.flush();
            RequestDispatcher dispatcher = request.getRequestDispatcher( “Now.jsp ”);
            dispatcher.include(request, response);
        %>
    </BODY>
</HTML>
```

Now.jsp를 include합니다

```
<%@page contentType=“text/html; charset=euc-kr”%>
<%@page import=“java.util.*”%>
<% GregorianCalendar now = new GregorianCalendar(); %>
 [현재의 시각] <%= String.format(“%TF %TT ”, now, now) %>
```



# Thank You !

뇌를 자극하는 JSP & Servlet

한빛미디어 Hanbit Media, Inc.