

# MySQL 데이터베이스 연결

# MySQL 데이터베이스 연결

## ■ Connection 객체 생성

&lt;%

```
String u_id = request.getParameter("userID");  
String u_pw = request.getParameter("userPW");  
String u_mail = request.getParameter("userMAIL");
```

```
String sql = "INSERT INTO members VALUES";  
sql += "(" + u_id + "', '" + u_pw + "', '" + u_mail + "'";
```

```
Class.forName("com.mysql.jdbc.Driver");
```

```
String url = "jdbc:mysql://localhost:3306/jspdb";  
String username = "root";  
String password = "1234";
```

```
Connection conn = DriverManager.getConnection(url, username, password);  
Statement sm = conn.createStatement();
```

:

☞ 다음과 같이 한 줄로 작성 가능

```
Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/jspdb", "root", "1234");
```

☆ 6p

# MySQL 데이터베이스 연결

## ■ Connection 객체 생성

- DriverManager 클래스의 static 매소드인 getConnection() 함수를 사용하여 데이터베이스와 연결
- 다음과 같이 3개의 파라미터를 입력받아서 데이터베이스 연결을 생성한 후  
연결 정보를 Connection 객체의 형태로 반환

(java.sql 패키지에 포함, java.sql.\* 패키지를 import)

메소드	반환 유형	입력 파라미터	
getConnection	java.sql.Connection	String url	접속할 데이터베이스 주소
		String user	사용자의 이름
		String password	사용자의 패스워드

```
Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/jspdb", "root", "1234");
```

## 쿼리 실행

- Statement 객체를 이용한 쿼리 실행

```
<%  
    String u_id = request.getParameter("userID");  
    String u_pw = request.getParameter("userPW");  
    String u_mail = request.getParameter("userMAIL");  
  
    String sql = "INSERT INTO members VALUES";  
    sql += "(" + u_id + "',''" + u_pw + "',''" + u_mail + "'')";  
  
    Class.forName("com.mysql.jdbc.Driver");  
  
    String url = "jdbc:mysql://localhost:3306/jspdb";  
    String username = "root";  
    String password = "1234";  
  
    Connection conn = DriverManager.getConnection(url, username, password);  
    Statement sm = conn.createStatement();  
  
    int count = sm.executeUpdate(sql);  
    if(count == 1){  
        out.println("회원가입 성공!");  
    }else{  
        out.println("회원가입 실패!");  
    }  
    sm.close();  
    conn.close();  
%>
```

# 쿼리 실행

- Statement 객체를 이용한 쿼리 실행

- Connection 객체가 생성되면,

SQL문을 데이터베이스로 전송하기 위해 Connection 객체의 createStatement() 함수 실행

👉 Statement 객체 생성

👉 Statement 클래스가 제공하는 함수 (SQL문에 따라 실행함수가 달라짐)

메소드	반환 유형	설명
<u>executeUpdate(String sql)</u>	int	<u>INSERT, DELETE, UPDATE 문을 실행하기 위한 함수</u> 이며, 실행 결과 변경된 레코드의 수를 반환
<u>executeQuery(String sql)</u>	java.sql.ResultSet	<u>SELECT 문을 실행하기 위한 함수</u> 로, 실행 결과로 얻어진 테이블 형태(Java에서는 ResultSet으로 표현)의 데이터를 반환

```
int count = sm.executeUpdate(sql);
```

# MySQL 데이터베이스 연동 ✨

## ■ 전체 과정 정리

데이터베이스 접속을 위해 DriverManager.getConnection()

↓

Connection 객체 생성

쿼리 실행 객체 생성을 위해 Connection 객체의 createStatement()

↓

Statement 객체 생성

Statement 객체를 이용한 쿼리 실행

```
executeUpdate(String sql)  
executeQuery(String sql)
```

## 자원 반납

- 객체의 close() 함수 호출

- 마지막으로 데이터베이스 접근 과정에서 사용된 Connection, Statement, ResultSet 객체를 close() 함수를 이용해 모두 닫아주면 된다.
- 명시적으로 닫아주지 않을 경우 메모리에 객체들이 계속 남아있게 되어 불필요한 가비지 컬렉션 유발

```
rs.close();  
stmt.close();  
conn.close();
```

## 👉 signup\_process.jsp 최종 코드

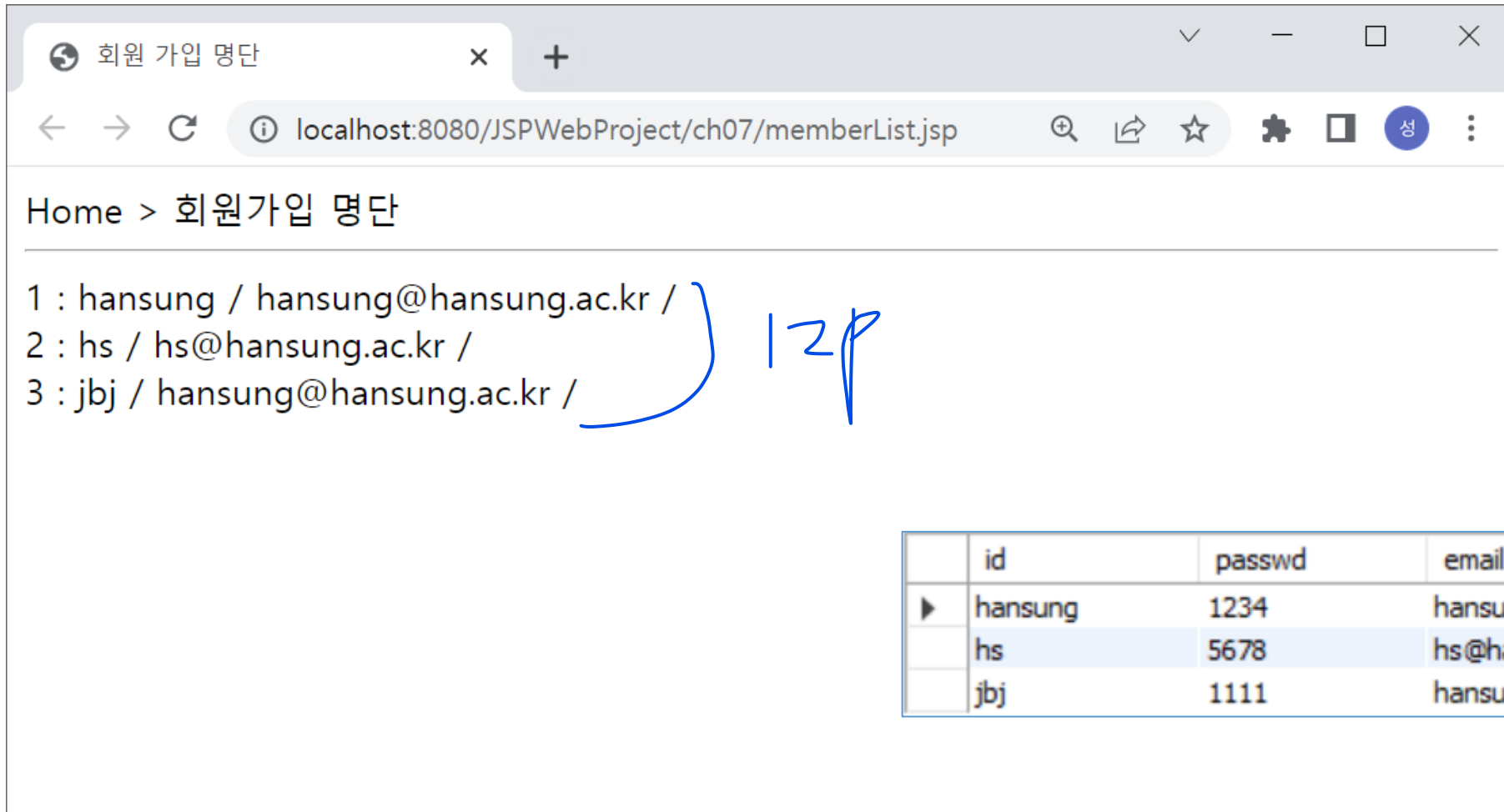
```
12 <%
13     String u_id = request.getParameter("userID");
14     String u_pw = request.getParameter("userPW");
15     String u_mail = request.getParameter("userMAIL");
16
17     String sql = "INSERT INTO members VALUES" + "(" + u_id + "," + u_pw + "," + u_mail + ")";
18
19     Class.forName("com.mysql.jdbc.Driver");
20
21     String url = "jdbc:mysql://localhost:3306/jspdb";
22     String username = "root";
23     String password = "1234";
24
25     Connection conn = DriverManager.getConnection(url, username, password);
26
27     // Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/jspdb", "root", "1234");
28
29     Statement sm = conn.createStatement();
30
31     int count = sm.executeUpdate(sql);
32     if(count == 1){
33         out.println("회원가입 성공!");
34     }else{
35         out.println("회원가입 실패!");
36     }
37     sm.close();
38     conn.close();
39 %>
```



# SELECT 구문 처리

# SELECT 구문 처리

- 실행 결과 (데이터베이스에서 전달받은 SELECT 구문 처리 결과 출력)



회원 가입 명단

localhost:8080/JSPWebProject/ch07/memberList.jsp

Home > 회원가입 명단

1 : hansung / hansung@hansung.ac.kr /  
2 : hs / hs@hansung.ac.kr /  
3 : jbj / hansung@hansung.ac.kr /

	id	passwd	email
▶	hansung	1234	hansung@hansung.ac.kr
	hs	5678	hs@hansung.ac.kr
	jbj	1111	hansung@hansung.ac.kr

# SELECT 구문 처리

- ch07 폴더에  
memberList.jsp 파일 생성

```
signup.jsp  signup_process.jsp  *memberList.jsp ×
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.sql.*" %>
4 <!DOCTYPE html>
5 <html>
6 <head>
7   <meta charset="UTF-8">
8   <title> 회원 가입 명단 </title>
9 </head>
10 <body>
11
12 <%
13
14   String driverName="com.mysql.jdbc.Driver";
15   Class.forName(driverName);
16
17   Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/jspdb", "root", "1234");
18
19   Statement sm = conn.createStatement();
20   ResultSet rs = sm.executeQuery("SELECT id, email FROM members");
21
22   out.print("Home > 회원가입 명단 <hr>");
23
24   String str = "";
25   int count = 1;
26
27   while(rs.next()){
28     str = count + " : " + rs.getString("id") + " / " + rs.getString("email") + " / " + "<br>";
29     out.print(str);
30     count++;
31   }
32
33   rs.close();
34   sm.close();
35   conn.close();
36 %>
37
38 </body>
39
40 </html>
```

	id	passwd	email
▶	hansung	1234	hansung@hansung.ac.kr
	hs	5678	hs@hansung.ac.kr
	jbj	1111	hansung@hansung.ac.kr

검색된 데이터

&lt;%

memberList.jsp

```
String driverName="com.mysql.jdbc.Driver";
Class.forName(driverName);
```

```
Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/jspdb", "root", "1234");
```

```
Statement sm = conn.createStatement();
ResultSet rs = sm.executeQuery("SELECT id, email FROM members");
```

```
out.print("Home > 회원가입 명단 <hr>");
```

```
String str = "";
```

```
int count = 1;
```

```
while(rs.next()) {
    str = count + " : " + rs.getString("id") + " / " + rs.getString("email") + " / " + "<br>";
    out.print(str);
    count++;
}
```

```
rs.close();
sm.close();
conn.close();
```

%&gt;

714

	id	password	email
	hansung	1234	hansung@hansung.ac.kr
	hs	1234	hs@hansung.ac.kr
	jbj	1234	hansung@hansung.ac.kr

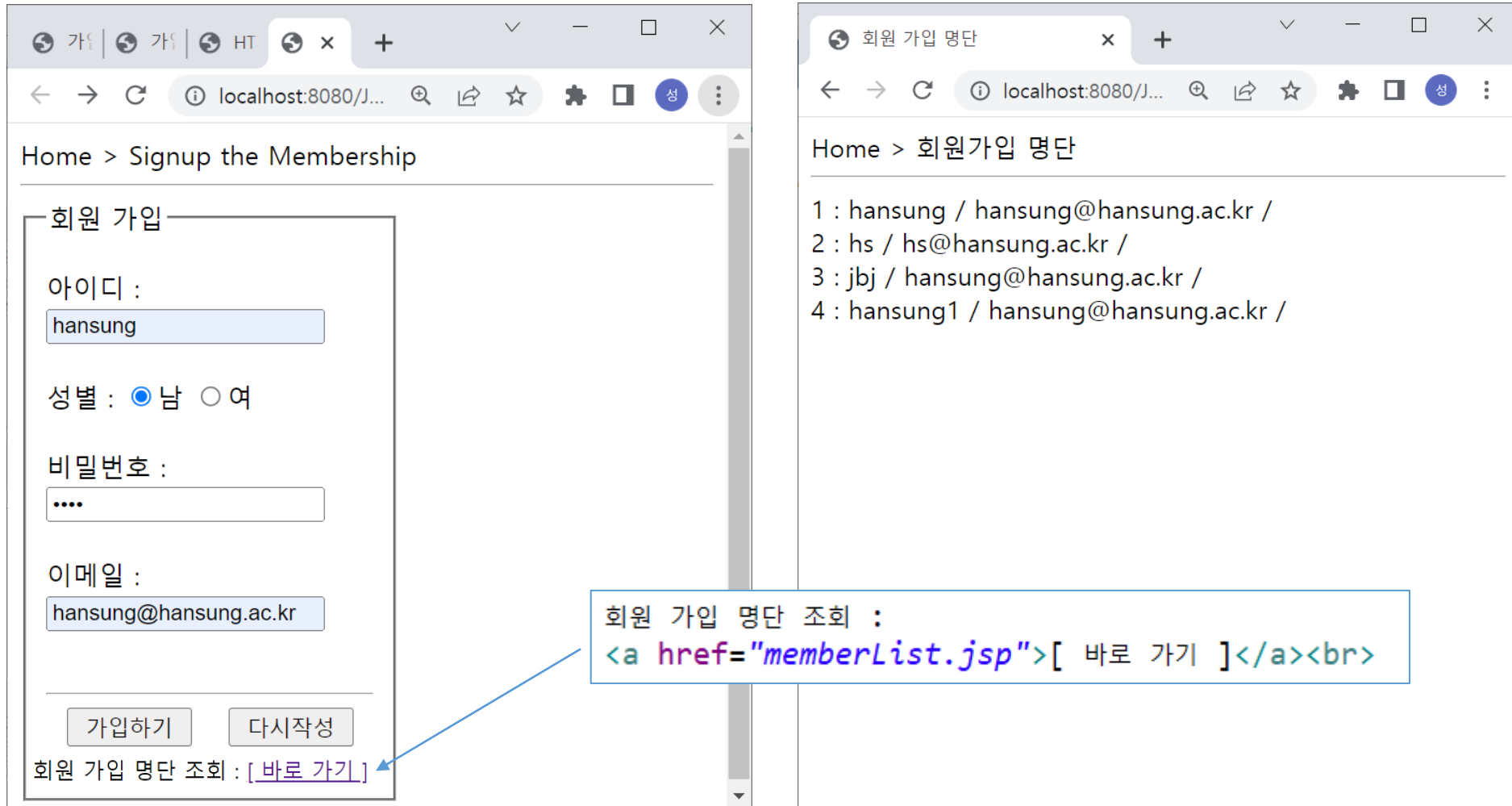
# 설명

## ▪ ResultSet 클래스의 주요 함수

메소드	반환 유형	설명
next()	boolean	현재 레코드를 가리키는 커서 <sup>Cursor</sup> 를 다음 레코드로 이동시킵니다. 다음 레코드가 존재하여 이동이 성공할 경우에는 true를, 그렇지 않는 경우 False를 반환합니다.
getString(String column)	String	커서가 가리키는 레코드 내 파라미터로 주어진 column 값을 String 타입으로 반환합니다.
getInt(String column)	int	커서가 가리키는 레코드 내 파라미터로 주어진 column 값을 int 타입으로 반환합니다.
first()	boolean	커서를 첫번째 레코드로 이동합니다.
last()	boolean	커서를 마지막 레코드로 이동합니다.

# 해보기

- 지난 시간 과제의 결과에  다음 <회원 가입 명단 조회> 링크를 추가하여 작성하시오.



Home > Signup the Membership

회원 가입

아이디 :  
hansung

성별 : ☒ 남 ☐ 여

비밀번호 :  
....

이메일 :  
hansung@hansung.ac.kr

가입하기    다시작성

회원 가입 명단 조회 : [\[바로 가기\]](#)

회원 가입 명단 조회 :

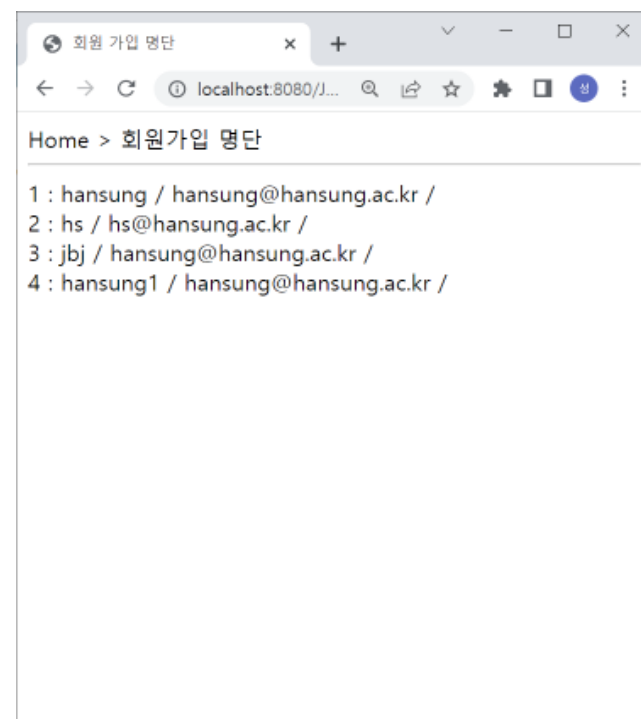
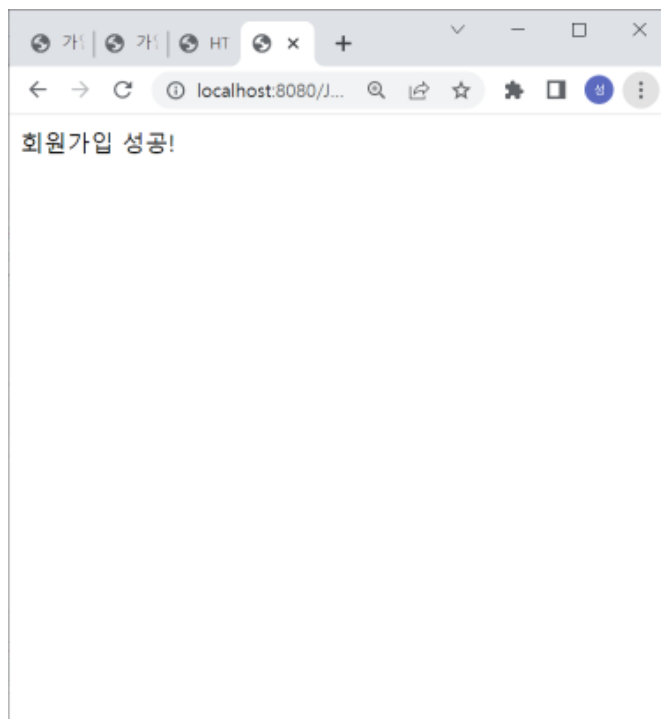
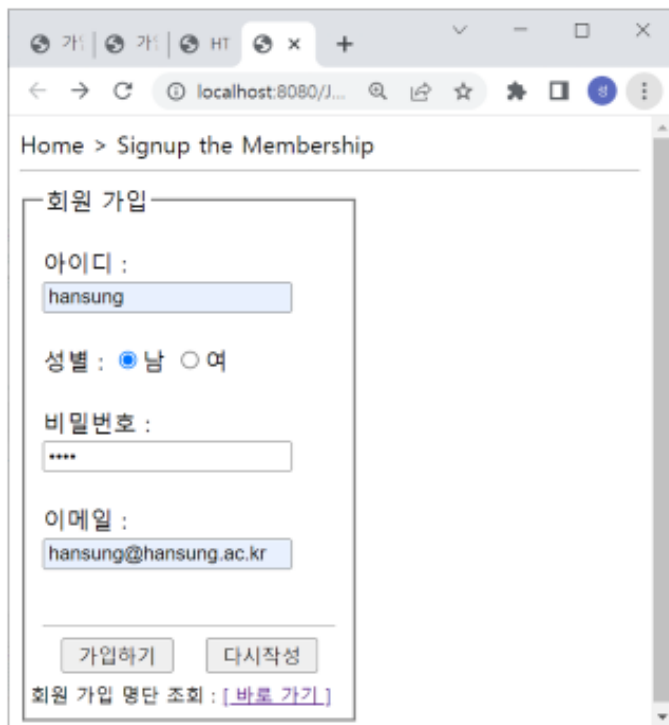
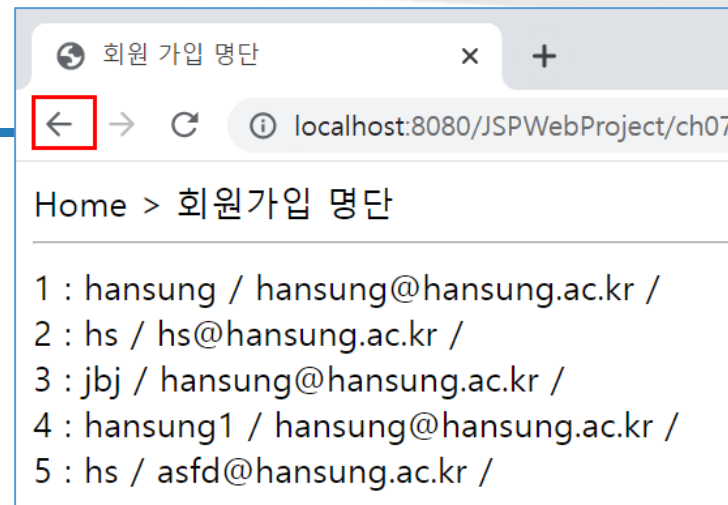
```
<a href="memberList.jsp">[ 바로 가기 ]</a><br>
```

Home > 회원가입 명단

1 : hansung / hansung@hansung.ac.kr /  
2 : hs / hs@hansung.ac.kr /  
3 : jbj / hansung@hansung.ac.kr /  
4 : hansung1 / hansung@hansung.ac.kr /

# 해보기

- 크롬에서 <뒤로 가기> 버튼을 눌러가며 결과 확인







 **T h a n k      y o u**

## **TECHNOLOGY**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Velit ex  
plicabo ipsum, labore sed tempora ratione asperiores des  
cenderat bore sed tempora rati jgert one bore sed tem!