

sql class day3

프레임워크 별 DB

JSP 이클립스로 hello world

spring 으로 hello world

spring boot 로 hello world

▼ 프레임워크 별 DB

프레임워크	DB
JSP	JDBC (Connection Pool)
Spring	JDBC (Mybatis)
Spring boot	JDBC (JPA)

▼ jsp, spring, spring boot 간략설명

jsp랑 spring 은 별개

그런데, spring boot는 spring에 플러스 알파된 개념!

근데 jsp에서 spring은 버전 하나 정도의 차이인데,

boot는 다른 느낌... 심지어 sql문을 사용하지 않음

오히려 jsp와 spring 간 간격보다

spring과 boot 간의 간격이 더 큼

jsp : 이층짜리 집

spring : 63층짜리 빌딩을 만드는 느낌

jsp 이전에 servlet이 존재했음

우리는 DB만 다룰 거기 때문에 jsp부터 다룰 예정이시라고 함

▼ 프레임워크란?

- 우리가 사이트를 만들듯이 햄버거가게를 만들고 싶다고 가정하면,
개인적으로는 햄버거가게를 위해 만들어야 할 게 너무 많음
근데, 프랜차이즈면? 훨씬 편리하지
프레임워크가 프랜차이즈와 같은 것
모든 게 준비되어 있고, 우리는 조립만 하면 되는 거지

사이트를 위해 일일이 만들기는 너무 힘든데,
프레임워크를 사용하면 만들어져 있는 것을 갖다붙이고 조립만 하면 되는 것



JSP의 DB연결방법 4가지

1. 각각의 페이지에서 직접 DB 사용

- 자원 소모가 심함
- 코드 중복 발생 (매 페이지에서 중복적으로 DB를 연결했다 끊었다 반복)

2. DB연결만 별도로 공용으로 사용 , file로 구성

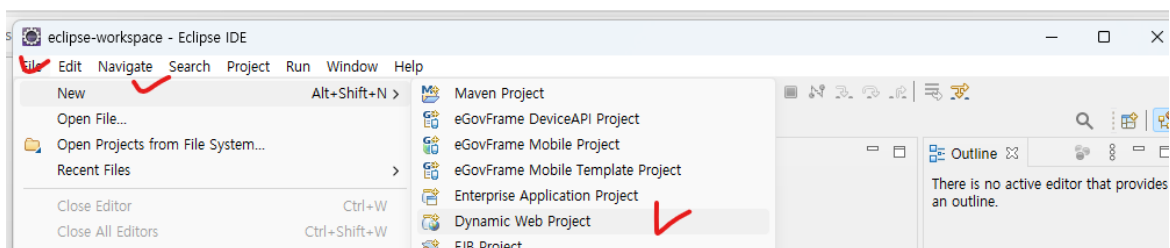
3. DB 연결만 별도로 공용으로 사용, 자바 class로 구현

4. **Connection Pool**로 DB 사용 [최종채택]

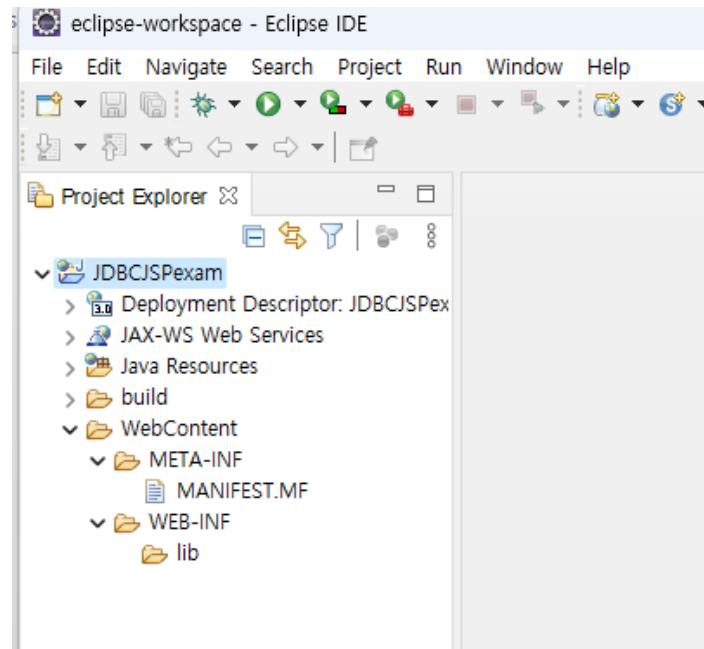
▼ JSP 이클립스로 hello world



JSP는 Dynamic Web Project 로 입장



next - next - next - finish !



Dynamic Web Project 로 만들었기 때문에,
Web Project 로서의 구조가 만들어짐

톰캣 설치 필요 (다운로드 설명)

톰캣이란? 로컬컴퓨터에서, 호스팅업체에서 돌리듯이 웹 서버에 돌릴 수 있게끔 하는 것
[Apache Tomcat® - Welcome!](#)



JSP는 Dynamic Web Project 로 입장

New Dynamic Web Project

Dynamic Web Project

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: JDBCJSPexam

Project location

☒ Use default location

Location: C:\Users\wso_y0\workspace\JDBCJSPexam Browse...

Target runtime

<None> New Runtime...

Dynamic web module version

3.0

Configuration

Default Configuration Modify...

The default configuration provides a good starting point. Additional facets can later be installed to add new functionality to the project.

EAR membership

☐ Add project to an EAR

EAR project name: EAR New Project...

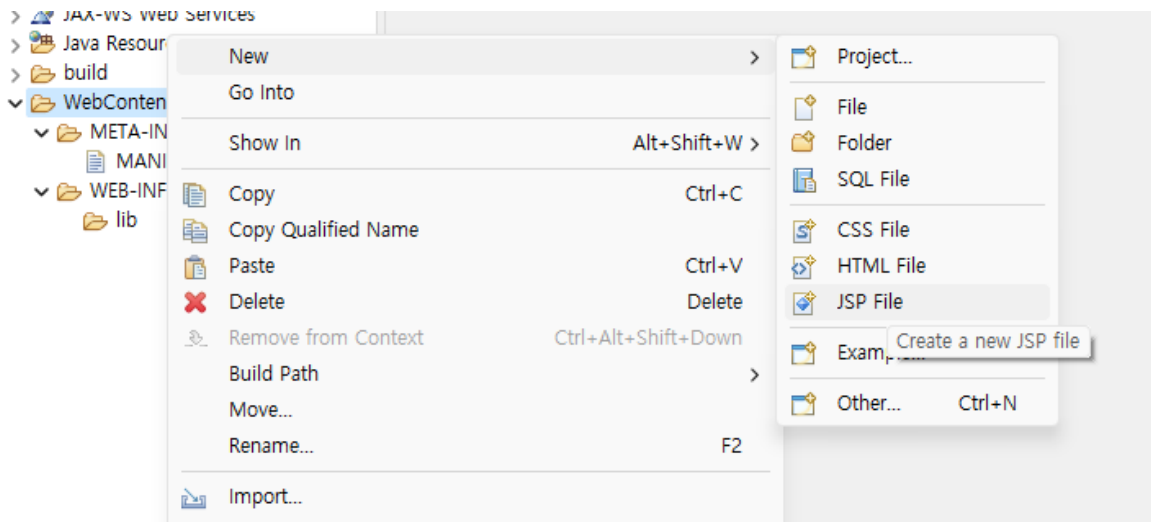
Working sets

☐ Add project to working sets New...

Working sets: Select...

? < Back Next > Finish Cancel

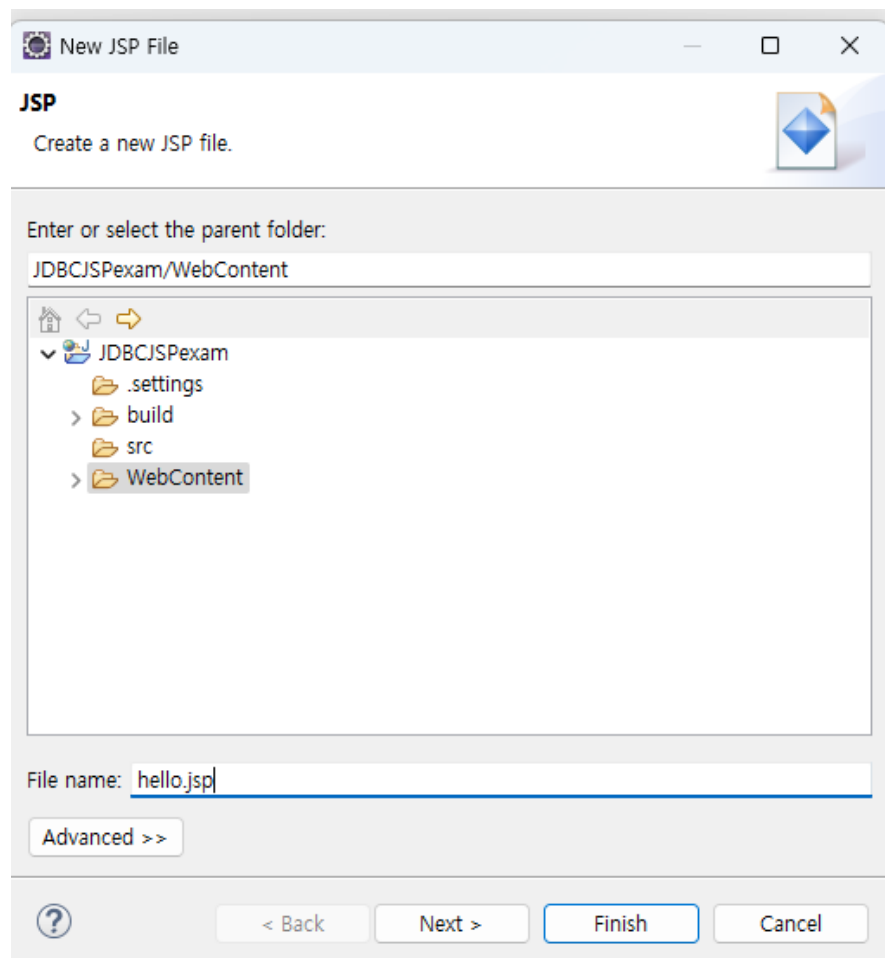
이제!



기본이 되는 폴더인 **WebContent** 밑에

[New] - [JSP File]

Web Content 폴더 밑에 생성해줘야, tomcat이 찾을 수 있음

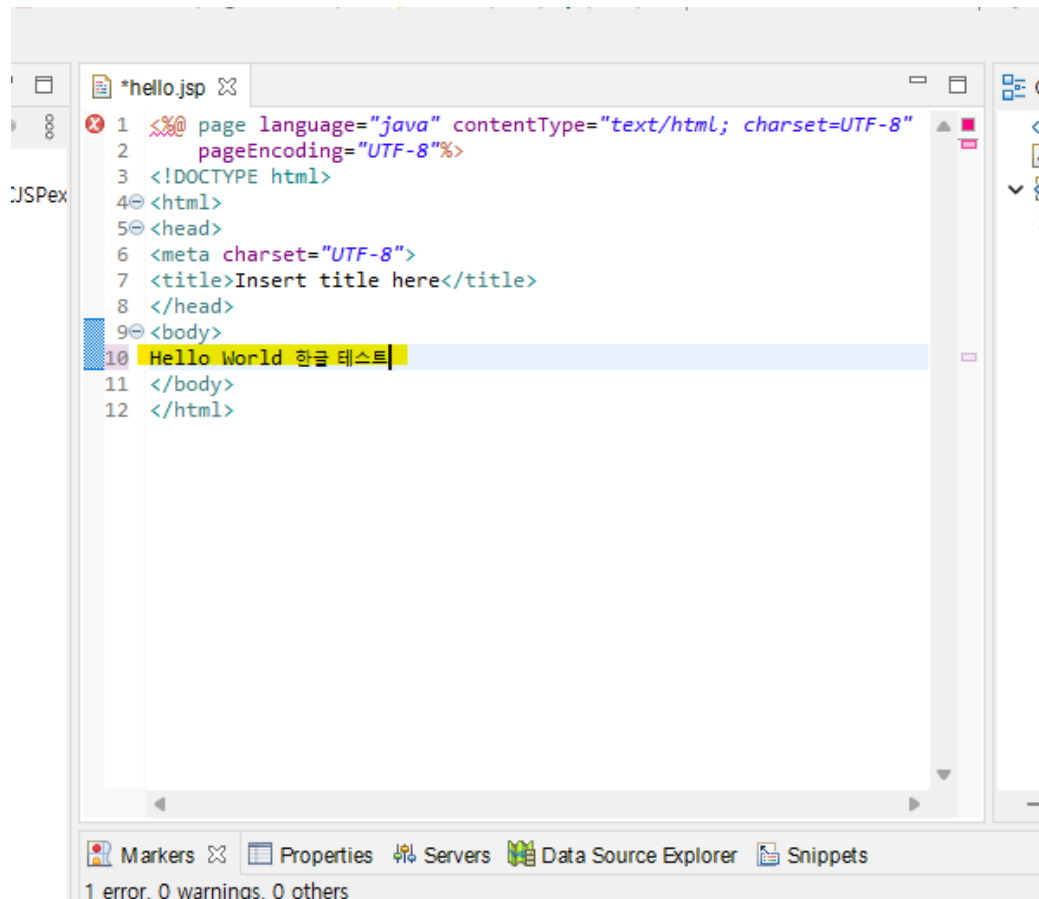


[finish]



```
hello.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8" %>
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6   <meta charset="UTF-8">
7   <title>Insert title here</title>
8 </head>
9 <body>
10
11 </body>
12 </html>
```

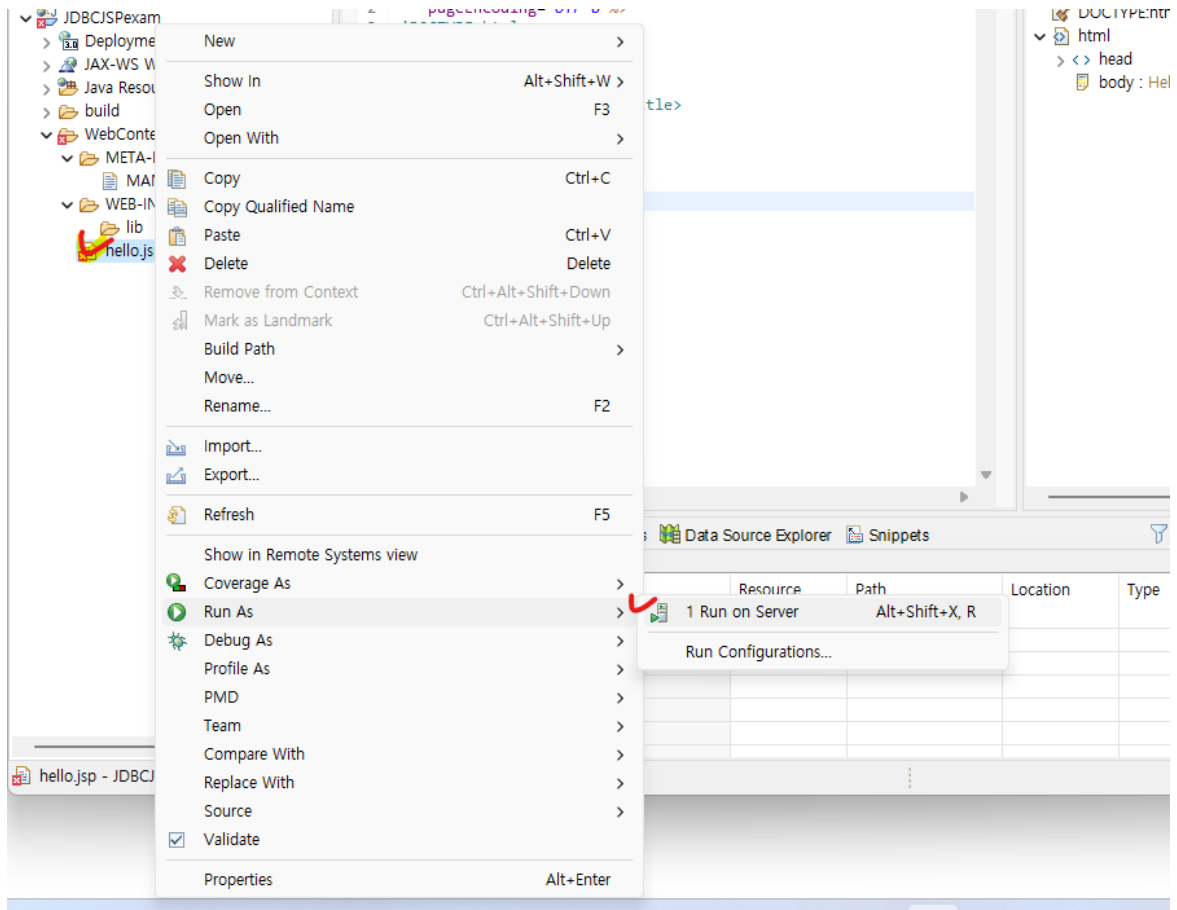
jsp파일로 생성했기 때문에, 기본 구조가 자동으로 생성됨



```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>Insert title here</title>
8 </head>
9 <body>
10 Hello World 한글 테스트
11 </body>
12 </html>
```

body부에 텍스트 작성

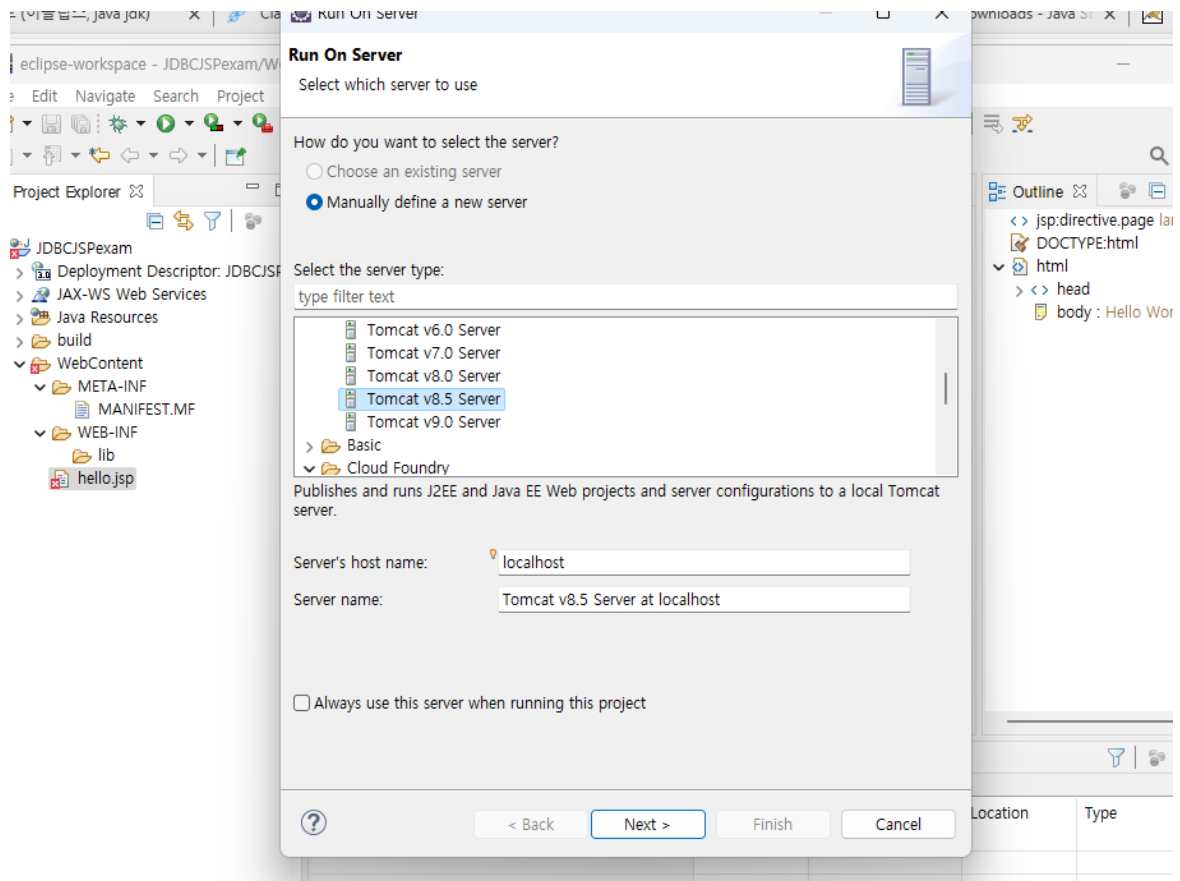
이제 저장 후, 이 페이지를 server에서 돌려보자

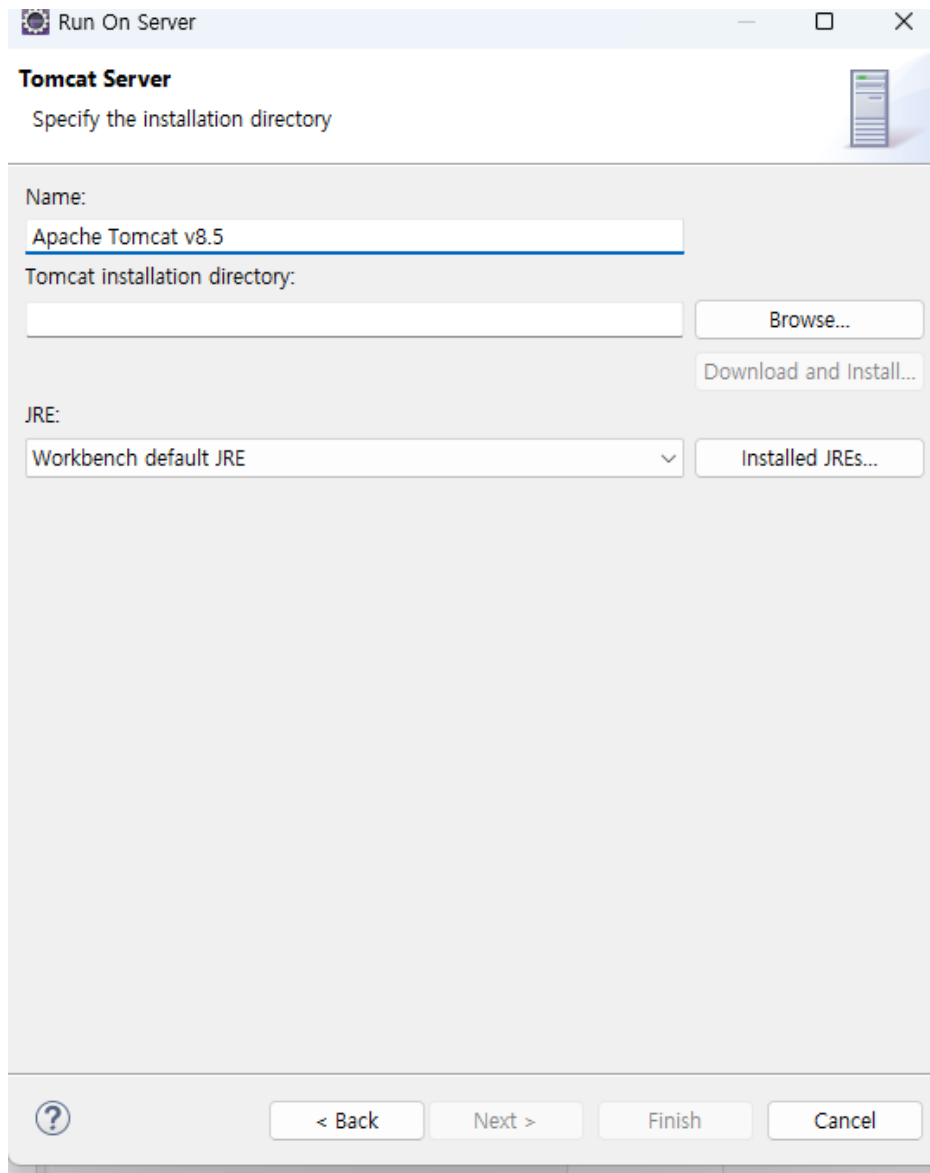


[New] - [Run As] - [Run on Server]

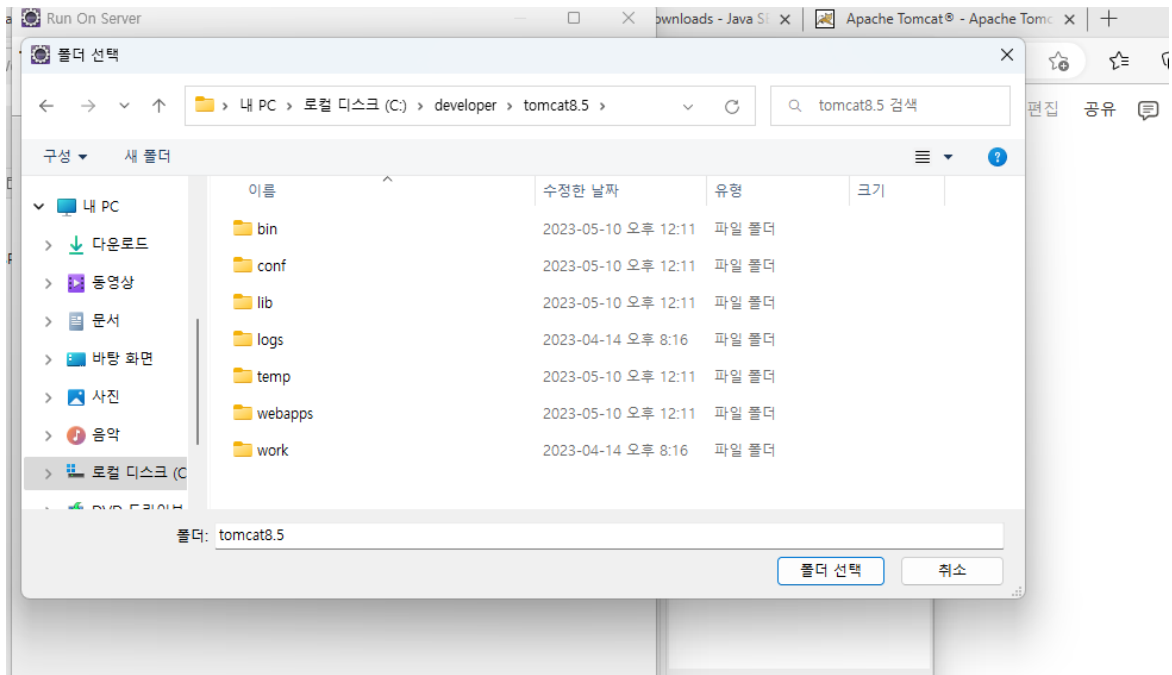
최초 실행 시, 톰캣 지정해줘야 함

설치했던 Tomcat 8.5 선택

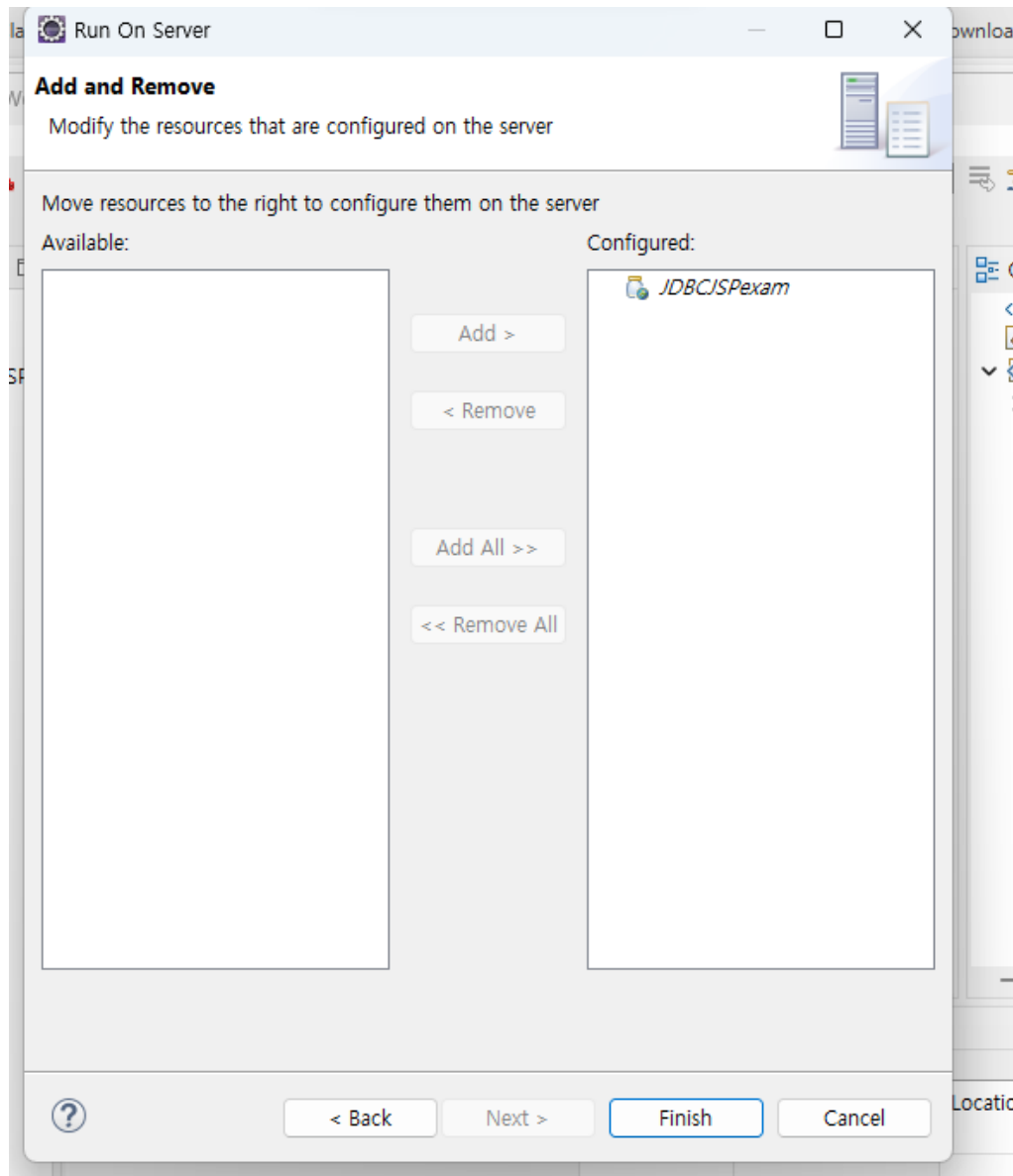




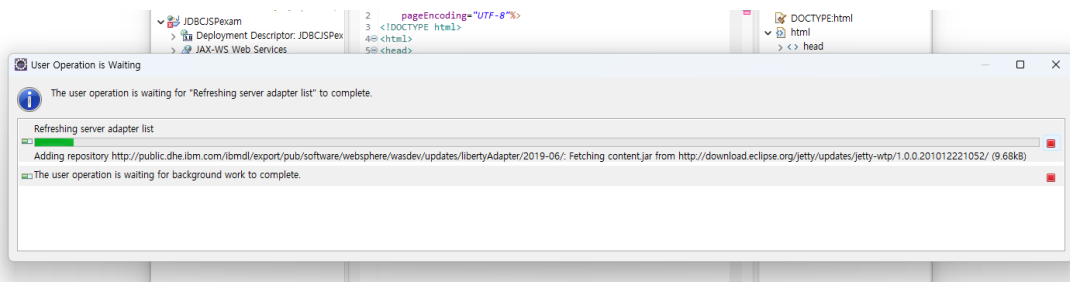
이전 단계에서 Tomcat 8.5는 선택했는데,
Tomcat8.5의 정확한 위치를 모르네? 지정해주자



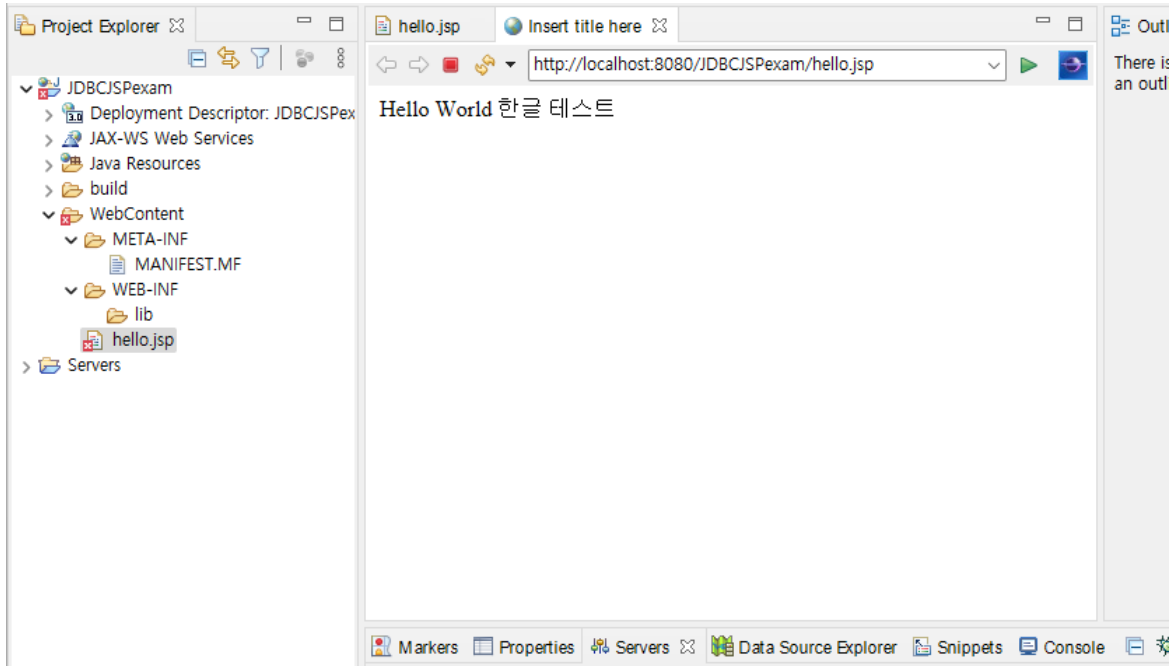
이전에 tomcat 설치했던 경로 ⇒ [C] - [developer] - [tomcat8.5] 였음
tomcat8.5 폴더까지만 잡아주고



[finish]

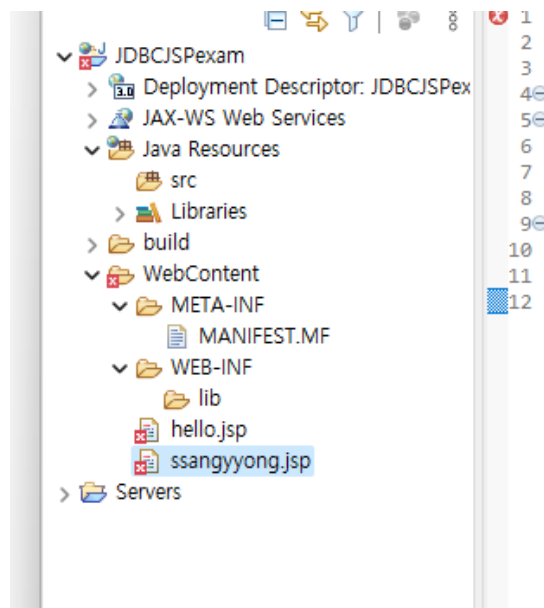


프랜차이즈 본사에 전화해서 필요한 거 이것저것 가져오는중~!



Run on Server 돌려보면? ⇒ 완료!

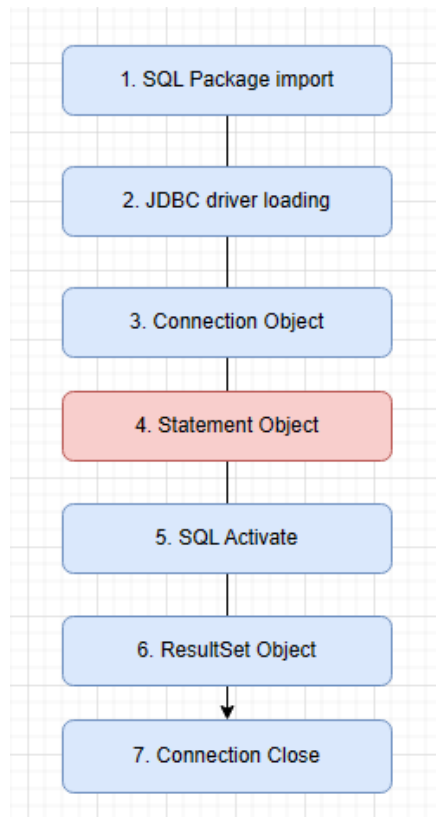
▼ JSP / Java 파일 생성 위치



jsp → Web Content
java → Java Resources

이제 DB 연결 내용 시작 !

>> DB 연결 7단계 <<



(3단계까지는 항상 동일함. 4단계가 가장 중요)

1. SQL Package import

⇒ 자바에서 sql을 쓸 수 있게끔 하는 sql이 담겨있는 패키지를 가져오는 것

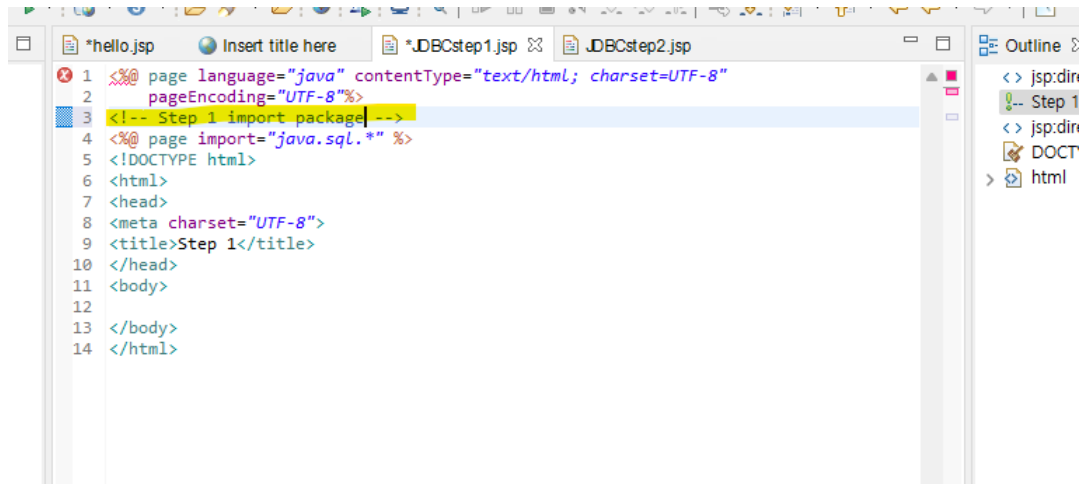
▼ 실습 화면

JDBCstep1.jsp

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.sql.*" %>
4 <!DOCTYPE html>
  
```

<%@ %> : 자바문법 데려오는 jsp 문법 中 선언의 의미



html 에서의 주석 : <!-- -->

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!-- Step 1 import package -->
<%@ page import="java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Step 1</title>
</head>
<body>

</body>
</html>
```

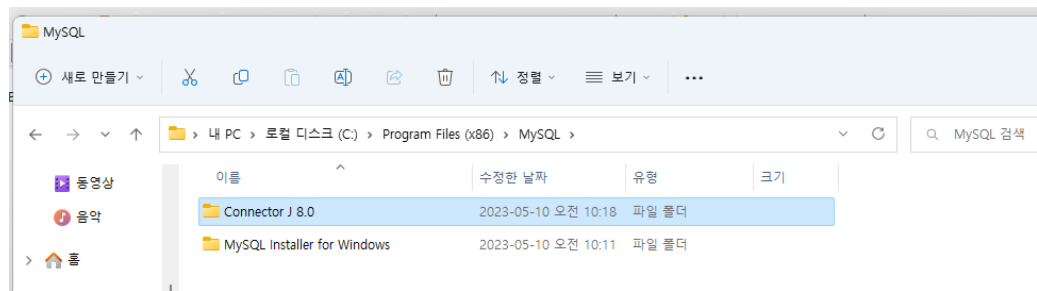
2. JDBC driver loading

⇒ sql 깔면서, JDBC/JDBC Connector/Work Bench를 설치했었음. 그 중 커넥터!

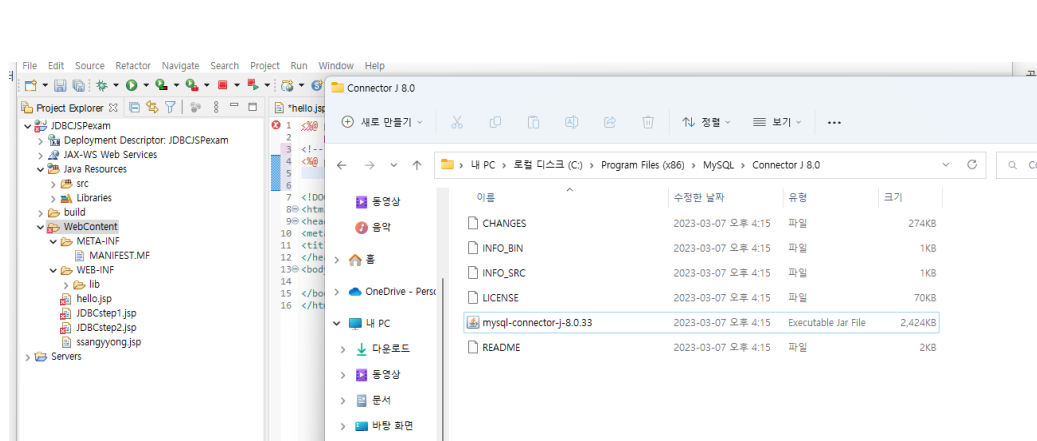
연결을 한 다음에 sql문을 던져야 함

▼ 실습 화면

MySQL 설치할 때의 경로에 driver 들어있음



[C] - [Program Files] - [MySQL]



MySQL 파일을 lib 폴더에 copy

```

15 <%
16     try {
17
18         Class.forName("com.mysql.jdbc.Driver");
19         out.print("드라이버 로드 성공 !!!<br>");
20
21     }catch(ClassNotFoundException err){
22
23         out.print("드라이버 로드 실패 !!!<br>" + err.getMessage());
24     }
25 %>
26

```

18번 코드 → 해당 드라이버를 읽어오라는 명령

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!-- Step 1 import package -->
<%@ page import="java.sql.*" %>

```

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Step 2</title>
</head>
<body>
<!-- Step 2 Load JDBC Driver -->

<%
    try {

        Class.forName("com.mysql.jdbc.Driver");
        out.print("드라이버 로드 성공 !!!<br>");

    }catch(ClassNotFoundException err){

        out.print("드라이버 로드 실패 !!!<br>" + err.getMessage());
    }
%>

```

3. Connection Object

⇒ 자바는 객체지향 언어임.

연결 객체를 만들어서, 커넥터를 타고 그 객체가 흐르도록 한 다음에 sql을 던지는 것

▼ 실습화면

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

- Connection 이라는 자료형의 conn 객체를 만들어줌.
—> step1 에서의 import문 덕분에 사용할 수 있는 것
- step2 에서 구동한 드라이버를 가져와서 내가 원하는 포트에 연결하는 것
- 커넥션 객체의 연결이 안되는 경우는 SQLException 에러 발생 (step2와 다름)

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!-- Step 1 import package -->
<%@ page import="java.sql.*" %>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Step 2</title>
</head>
<body>
<!-- Step 2 Load JDBC Driver -->

<%
    try {

        Class.forName("com.mysql.jdbc.Driver");
        out.print("드라이버 로드 성공 !!!<br>");

    }catch(ClassNotFoundException err){

        out.print("드라이버 로드 실패 !!!<br>" + err.getMessage());
    }
%>

<!-- Step 3 Connection Object -->

<%

    Connection conn = null;
    try {

        conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/", "root", "0000");
        out.print("연결객체 생성 성공 !!!<br>");

    }catch(SQLException err){

        out.print("연결객체 생성 실패 !!!<br>" + err.getMessage());
    }
%>

</body>
</html>

```

4. Statement Object [제일 중요]

⇒ Statement 는 SQL문을 의미

⇒ 구문을 만들지만 했을 뿐, 실행한 것은 아님

▼ 실습화면

```

<!-- Step 4 Connection Object -->

<%
    try {

        String sql = "CREATE DATABASE test";
        PreparedStatement pstmt = conn.prepareStatement(sql);
        out.print("구문 생성 성공 !!!<br>");

    } catch (SQLException err) {

```

test 라는 데이터베이스를 생성하겠다는 sql 문장

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!-- Step 1 import package -->
<%@ page import="java.sql.*" %>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Step 2</title>
</head>
<body>
<!-- Step 2 Load JDBC Driver -->

<%
    try {

        Class.forName("com.mysql.jdbc.Driver");
        out.print("드라이버 로드 성공 !!!<br>");

    } catch (ClassNotFoundException err) {

        out.print("드라이버 로드 실패 !!!<br>" + err.getMessage());
    }
%>

<!-- Step 3 Connection Object -->

<%

    Connection conn = null;
    try {

        conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/", "root", "0000");
        out.print("연결 객체 생성 성공 !!!<br>");

    } catch (SQLException err) {

        out.print("연결 객체 생성 실패 !!!<br>" + err.getMessage());
    }
%>

<!-- Step 4 Connection Object -->

<%

```

```

try {

    String sql = "CREATE DATABASE test";
    PreparedStatement pstmt = conn.prepareStatement(sql);
    out.print("구문 생성 성공 !!!<br>");

} catch (SQLException err){

    out.print("구문 생성 실패 !!!<br>" + err.getMessage());

}
%>

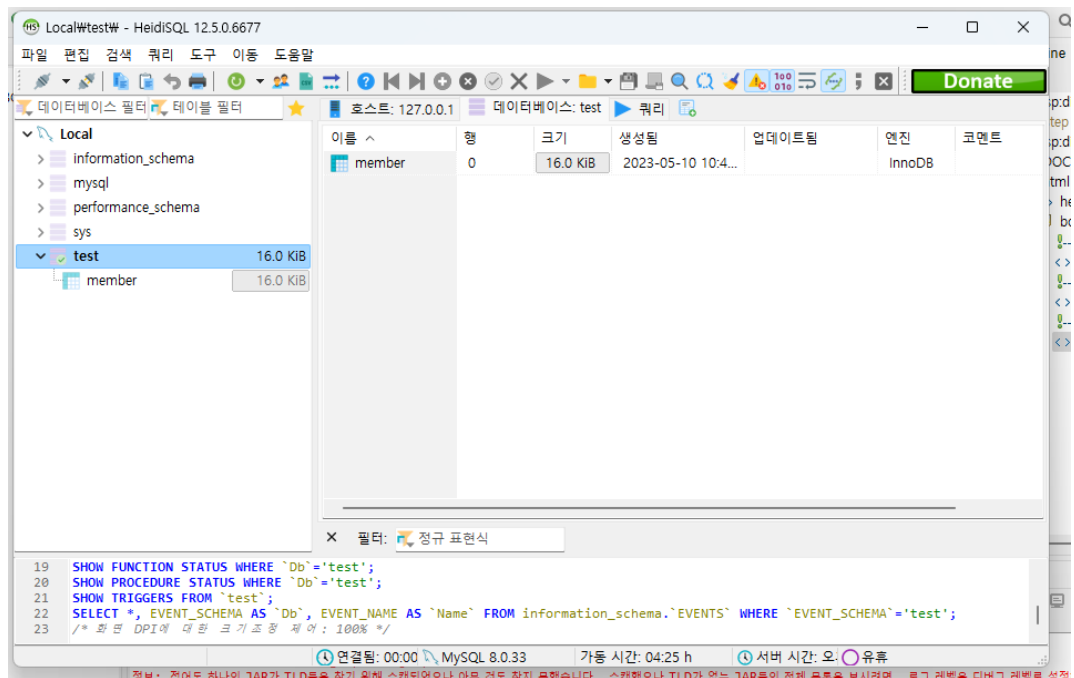
</body>
</html>

```

5. SQL Activate

⇒ 실제로 SQL문을 던져주는 것

▼ 실습화면



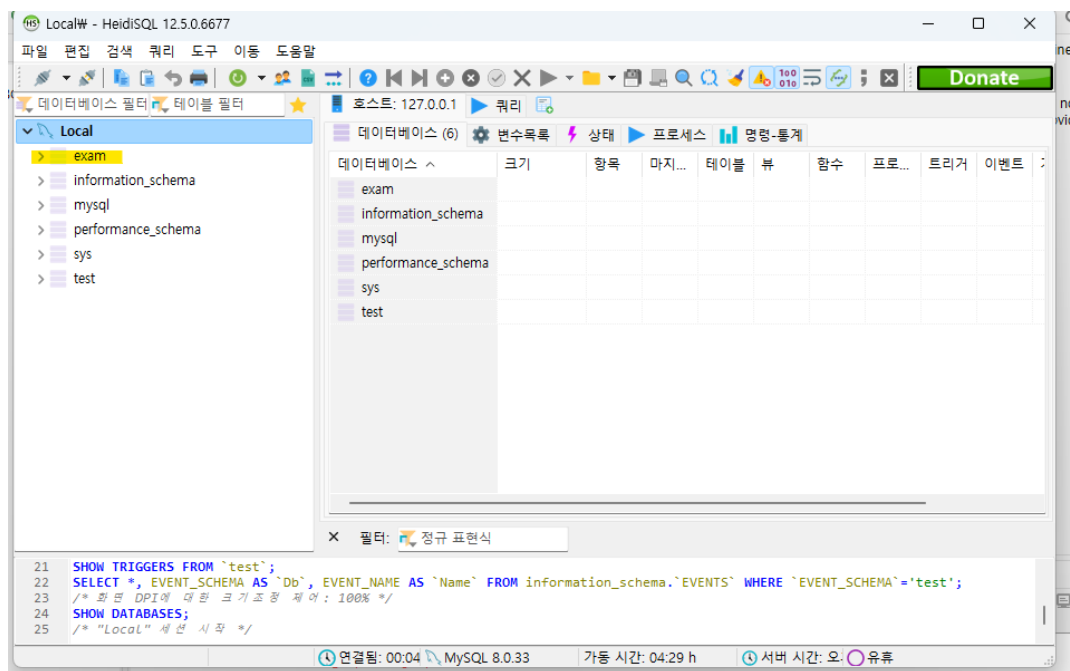
원래 헤이디에 없었는데 !

```

>/
58 <!-- Step 5 SQL Activate -->
59
60 <%
61
62     pstmt.executeUpdate();
63     out.print("DB 생성 성공 !!!<br>");
64
65
66 %>
67

```

pstmt 를 실행해주었더니



생김!!

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!-- Step 1 import package -->
<%@ page import="java.sql.*" %>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Step 5</title>
</head>

```

```

<body>
<!-- Step 2 Load JDBC Driver -->

<%
    try {

        Class.forName("com.mysql.jdbc.Driver");
        out.print("드라이버 로드 성공 !!!<br>");

    }catch(ClassNotFoundException err){

        out.print("드라이버 로드 실패 !!!<br>" + err.getMessage());
    }
%>

<!-- Step 3 Connection Object -->

<%
    Connection conn = null;
    try {

        conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/", "root", "0000");
        out.print("연결 객체 생성 성공 !!!<br>");

    }catch(SQLException err){

        out.print("연결 객체 생성 실패 !!!<br>" + err.getMessage());
    }
%>

<!-- Step 4 Statement Object -->

<%
    String sql = "CREATE DATABASE exam";
    PreparedStatement pstmt = conn.prepareStatement(sql);
    out.print("구문 생성 성공 !!!<br>");

%>

<!-- Step 5 SQL Activate -->

<%
    pstmt.executeUpdate();
    out.print("DB 생성 성공 !!!<br>");

%>

</body>
</html>

```

6. **ResultSet Object** (CRUD 중 R만 ResultSet 존재)

⇒ CRUD 중 C, U, D 는 ResultSet이 없고, R은 ResultSet이 존재

▼ 실습화면

현재는 6번이 필요하지 않지만, 명목상 표현!

7. Connection Close

⇒ 연결을 해서 작업을 한 후, close 해주는 것. (끊어줌)

※ Java8 부터는 close 하지 않아도 괜찮음

▼ 실습화면

```
71 <!-- Step 7 Connection Close 자바 1.8부터는 생략 가능 -->
72
73 pstmt.Close();
74 conn.Close();
75
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<!-- Step 1 import package -->
<%@ page import="java.sql.*" %>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Step 7</title>
</head>
<body>

<!-- Step 2 Load JDBC Driver -->

<%
    try {

        Class.forName("com.mysql.jdbc.Driver");
        out.print("드라이버 로드 성공 !!!<br>");

    }catch(ClassNotFoundException err){

        out.print("드라이버 로드 실패 !!!<br>" + err.getMessage());
    }
%>

<!-- Step 3 Connection Object -->

<%
```

```

Connection conn = null;
try {

    conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/", "root", "0000");
    out.print("연결 객체 생성 성공 !!!<br>");

} catch(SQLException err){

    out.print("연결 객체 생성 실패 !!!<br>" + err.getMessage());
}
%>

<!-- Step 4 Statement Object -->

<%

    String sql = "CREATE DATABASE exam";
    PreparedStatement pstmt = conn.prepareStatement(sql);
    out.print("구문 생성 성공 !!!<br>");

%>

<!-- Step 5 SQL Activate -->

<%

    pstmt.executeUpdate();
    out.print("DB 생성 성공 !!!<br>");

%>

<!-- Step 6 ResultSet Object -->
<!-- 디비 생성은 가지고 오는 값이 없으므로 생략 -->

<!-- Step 7 Connection Close 자바 1.8부터는 생략 가능 -->

    pstmt.close();
    conn.close();

</body>
</html>

```

[과제]

JDBC0 이라는 이름으로, 깔끔하게 하나 만들어보기 → user 라는 데이터베이스 만들 것 !

⇒ 내가 하던 JDBC0 에러남.. 찾아보기! 500번 에러

이제 테이블을 만들어볼 것

▼ 테이블 생성 실습

```
String sql = "CREATE TABLE board("
    + "bno int,"
    + "btitle varchar(50),"
    + "bcontent varchar(50))";
```

- java / sql 각각에서는 줄바꿈 상관 없는데, java에서 sql문 작성할 때는 영향 있음
-

```
String sql = "CREATE TABLE board("
    + "bno int,"
    + "btitle varchar(50),"
    + "bcontent varchar(50))";

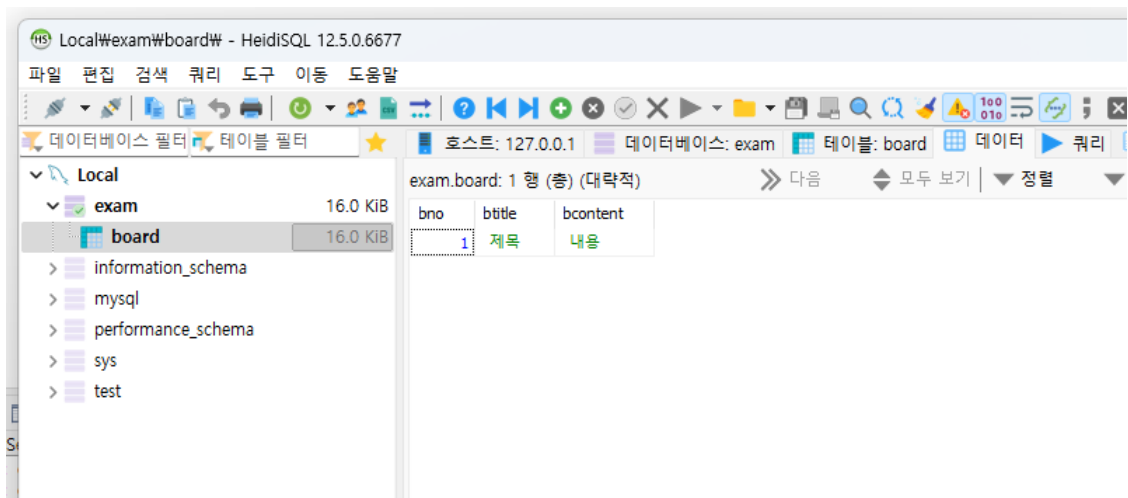
PreparedStatement pstmt = conn.prepareStatement(sql);
out.print("구문 생성 성공 !!!<br>");
```

▼ 테이블 속 데이터 생성 (FM / 자리표시자 생성) - C

첫 번째 방법) FM 방식

```
String sql = "INSERT INTO board values(1, '제목', '내용')";

PreparedStatement pstmt = conn.prepareStatement(sql);
out.print("삽입 성공 !!!<br>");
```

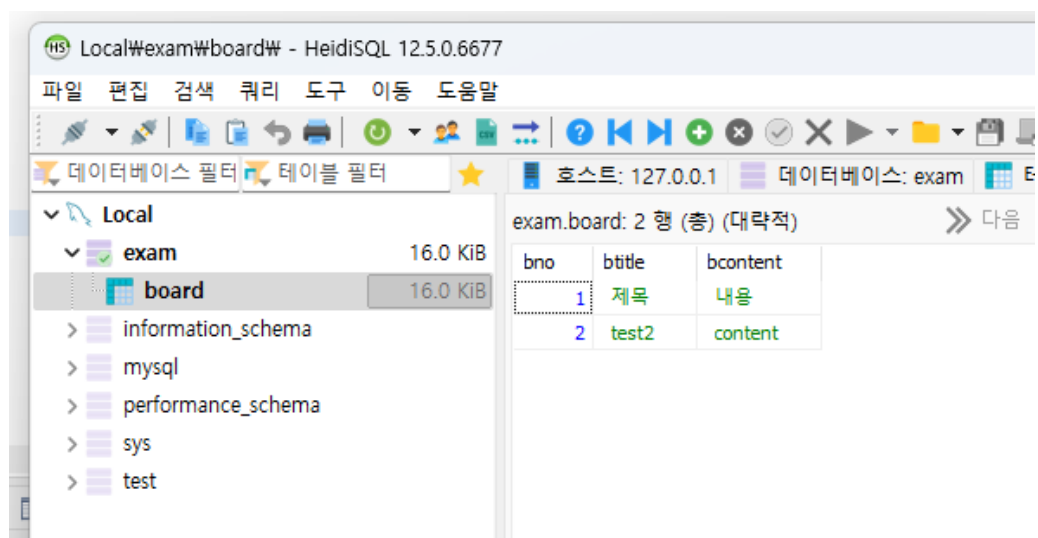


- 실제로 데이터 값을 넣을 때는, 속성 종류가 많을 텐데

그러면 몇십 줄을 다 넣어야 함

두 번째 방법) **자리 표시자** 만들어서 객체에 집어넣기

```
<%  
  
String sql = "INSERT INTO board values(?, ?, ?)";  
  
PreparedStatement pstmt = conn.prepareStatement(sql);  
    pstmt.setInt(1,2);  
    pstmt.setString(2,"test2");  
    pstmt.setString(3,"content");  
out.print("삽입 성공 !!!<br>");  
  
%>
```



▼ 테이블 속 데이터 삭제 - D

```

<%
String sql = "DELETE FROM board where bno=?";

PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setInt(1,2);

%>

<%
pstmt.executeUpdate();
out.print("삭제 성공 !!!<br>");
%>

```

```

<%
String sql = "DELETE FROM board where bno=?";

PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setInt(1,2);
%>

```

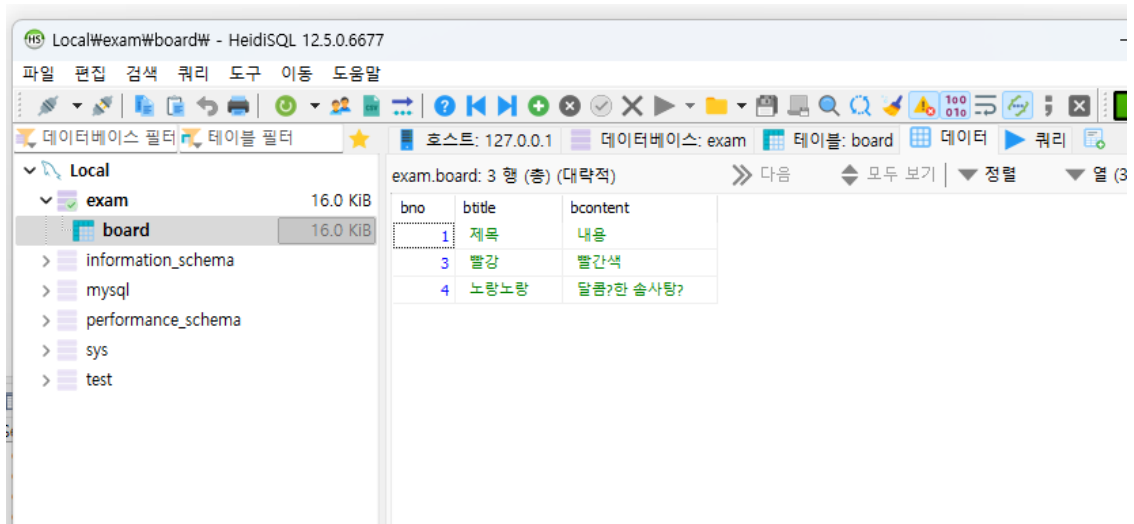
before

데이터베이스 필터 테이블 필터 ★ 호스트: 127.0.0.1 데이터베이스: exam 테이블

exam.board: 4 행 (총) (대략적) >> 다음		
bno	btitle	bcontent
1	제목	내용
2	test2	content
3	빨강	빨간색
4	노랑노랑	달콤?한 솜사탕?

Local
 exam 16.0 KiB
 board 16.0 KiB
 information_schema
 mysql
 performance_schema
 sys
 test

after



▼ 테이블 속 데이터 업데이트 - U

```
<%

String sql = "UPDATE board SET btitle = ? WHERE bno = ?";

PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setString(1, "변경테스트");
pstmt.setInt(2,3);

%>
```

```
<%

String sql = "UPDATE board SET btitle = ? WHERE bno = ?";

PreparedStatement pstmt = conn.prepareStatement(sql);
pstmt.setString(1, "변경테스트");
pstmt.setInt(2,3);

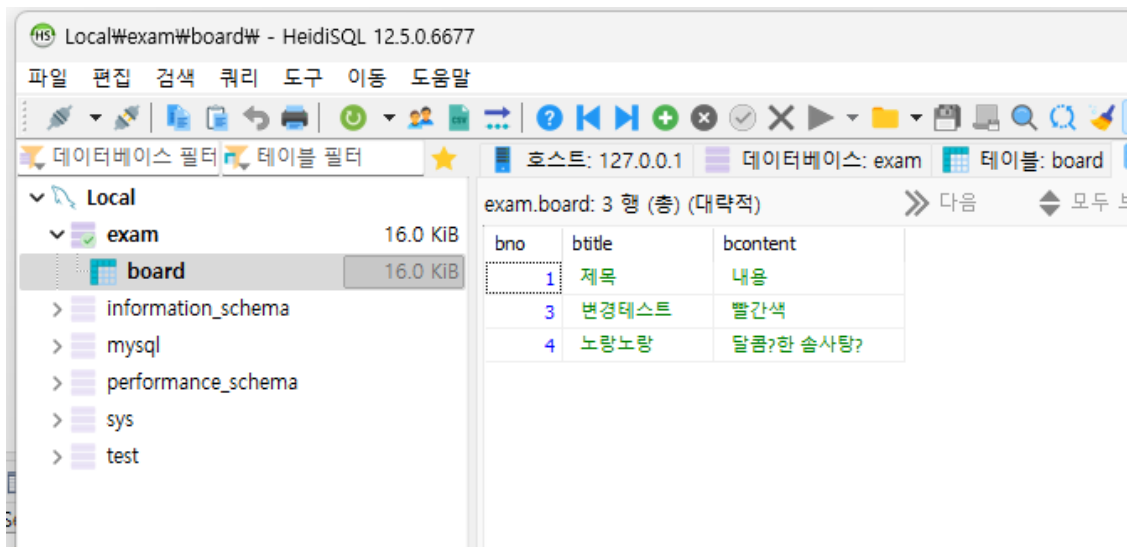
%>

<%

pstmt.executeUpdate();
out.print("수정 성공 !!!<br>");

%>
```

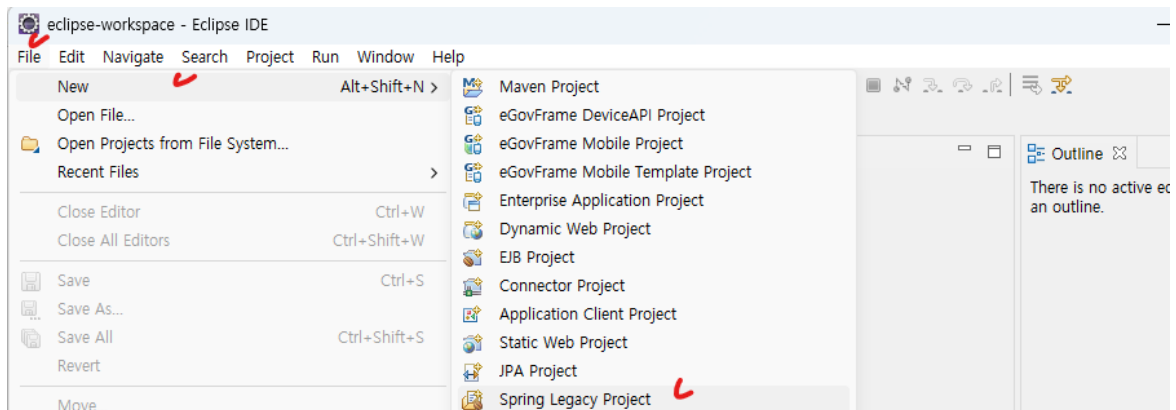
after



▼ spring 으로 hello world



spring은 spring legacy project 로 입장



▼ spring boot 로 hello world



spring boot는 spring legacy project 로 입장

