22장 회원제 게시판 구현2: 게시판 기능

22.1 게시판 기능 목록

■ 구현할 게시판 기능은 다음과 같다.
- 게시글 등록
- 게시글 목록 조회
- 게시글 내용 조회
- 게시글 수정

22.2 예제를 위한 테이블 생성

(1) ddl.sql (게시판 관련 테이블)

[board/sql/ddl.sql]

```
01    create table board.article (
02        article_no int auto_increment primary key,
03        writer_id varchar(50) not null,
04        writer_name varchar(50) not null,
05        title varchar(255) not null,
06        regdate datetime not null,
07        moddate datetime not null,
08        read_cnt int
09    ) engine=InnoDB default character set = utf8;
10
11    create table board.article_content (
12        article_no int primary key,
13        content text
14    ) engine=InnoDB default character set = utf8;
```

22.3 Writer, Article, ArticleContent 구현

(1) Writer.java

[board/src/article/model/Writer.java]

```
01    package article.model;
02
03    public class Writer {
04
05              private String id;
06              private String name;
07
08              public Writer(String id, String name) {
09                        this.id = id;
10                        this.name = name;
11              }
12
13              public String getId() {
14                        return id;
15              }
16
17              public String getName() {
18                        return name;
```

```
19              }
20
21      }
```

(2) Article.java

[board/src/article/model/Article.java]

```
01      package article.model;
02
03      import java.util.Date;
04
05      public class Article {
06
07              private Integer number;
08              private Writer writer;
09              private String title;
10              private Date regDate;
11              private Date modifiedDate;
12              private int readCount;
13
14              public Article(Integer number, Writer writer, String title,
15                                      Date regDate, Date modifiedDate, int readCount) {
16                      this.number = number;
17                      this.writer = writer;
18                      this.title = title;
19                      this.regDate = regDate;
20                      this.modifiedDate = modifiedDate;
21                      this.readCount = readCount;
22              }
23
24              public Integer getNumber() {
25                      return number;
26              }
27
28              public Writer getWriter() {
29                      return writer;
30              }
31
32              public String getTitle() {
33                      return title;
34              }
35
36              public Date getRegDate() {
37                      return regDate;
38              }
39
40              public Date getModifiedDate() {
41                      return modifiedDate;
42              }
43
44              public int getReadCount() {
45                      return readCount;
46              }
47
48      }
```

(3) ArticleContent.java

[board/src/article/model/ArticleContent.java]

```
01    package article.model;
02
03    public class ArticleContent {
04
05              private Integer number;
06              private String content;
07
08              public ArticleContent(Integer number, String content) {
09                        this.number = number;
10                        this.content = content;
11              }
12
13              public Integer getNumber() {
14                        return number;
15              }
16
17              public String getContent() {
18                        return content;
19              }
20
21    }
```

22.4 게시판 쓰기 기능 구현

(1) 게시글 쓰기 기능의 요구사항

- 게시글을 쓰기 위한 폼은 제목과 내용을 제공한다.
- 폼에 제목을 입력하지 않았으면 다시 폼을 보여준다.
- 로그인 한 사용자만 게시글 쓰기를 할 수 있다.

(2) 기능을 구현하기 위해 만들 코드 구성

```
WriterArticleHandler    → WriteArticleService  → ArticleDao        → board DB
        ↓                   WriteRequest           ArticleContentDao
writeArticleForm.jsp,
writeArticleSuccess.jsp
```

(3) 각 클래스와 JSP 코드의 기능

- ArticleDao 클래스와 ArticleContentDao 클래스 : article 테이블과 article_content 테이블에 데이터를 삽입하는 기능을 제공한다.
- WriteArticleService 클래스 : 새로운 게시글 쓰기 기능을 제공한다. 두 DAO 클래스를 이용해서 기능을 구현한다.
  - WriteRequest 클래스 : 신규 게시글을 등록할 때 필요한 데이터를 제공한다.
- WriteArticleHandler 클래스 : 사용자의 게시글 쓰기 요청을 처리한다.

22.4.1 ArticleDao와 ArticleContentDao의 데이터 추가 기능 구현

(1) ArticleDao.java

[board/src/article/dao/ArticleDao.java]

```
01   package article.dao;
02
03   import java.sql.Connection;
04   import java.sql.PreparedStatement;
05   import java.sql.ResultSet;
06   import java.sql.SQLException;
07   import java.sql.Statement;
08   import java.sql.Timestamp;
09   import java.util.ArrayList;
10   import java.util.Date;
11   import java.util.List;
12
13   import article.model.Article;
14   import article.model.Writer;
15   import jdbc.JdbcUtil;
16
17   public class ArticleDao {
18
19            public Article insert(Connection conn, Article article) throws SQLException {
20                     PreparedStatement pstmt = null;
21                     Statement stmt = null;
22                     ResultSet rs = null;
23                     try {
24                              pstmt = conn.prepareStatement("insert into article "
25                                       + "(writer_id, writer_name, title, regdate, moddate,
26   read_cnt) "
27                                       + "values (?,?,?,?,?,0)");
28                              pstmt.setString(1, article.getWriter().getId());
29                              pstmt.setString(2, article.getWriter().getName());
30                              pstmt.setString(3, article.getTitle());
31                              pstmt.setTimestamp(4, toTimestamp(article.getRegDate()));
32                              pstmt.setTimestamp(5, toTimestamp(article.getModifiedDate()));
33                              int insertedCount = pstmt.executeUpdate();
34
35                              if (insertedCount > 0) {
36                                       stmt = conn.createStatement();
37                                       rs = stmt.executeQuery("select last_insert_id() from article");
38                                       if (rs.next()) {
39                                                Integer newNo = rs.getInt(1);
40                                                return new Article(newNo,
41                                                         article.getWriter(),
42                                                         article.getTitle(),
43                                                         article.getRegDate(),
44                                                         article.getModifiedDate(),
45                                                         0);
46                                       }
47                              }
48                              return null;
49                     } finally {
50                              JdbcUtil.close(rs);
51                              JdbcUtil.close(stmt);
52                              JdbcUtil.close(pstmt);
53                     }
54            }
55
56            private Timestamp toTimestamp(Date date) {
57                     return new Timestamp(date.getTime());
```

```
58              }
59
60              public int selectCount(Connection conn) throws SQLException {
61                        Statement stmt = null;
62                        ResultSet rs = null;
63                        try {
64                                  stmt = conn.createStatement();
65                                  rs = stmt.executeQuery("select count(*) from article");
66                                  if (rs.next()) {
67                                            return rs.getInt(1);
68                                  }
69                                  return 0;
70                        } finally {
71                                  JdbcUtil.close(rs);
72                                  JdbcUtil.close(stmt);
73                        }
74              }
75
76              public List<Article> select(Connection conn, int startRow, int size) throws SQLException {
77                        PreparedStatement pstmt = null;
78                        ResultSet rs = null;
79                        try {
80                                  pstmt = conn.prepareStatement("select * from article " +
81                                                      "order by article_no desc limit ?, ?");
82                                  pstmt.setInt(1, startRow);
83                                  pstmt.setInt(2, size);
84                                  rs = pstmt.executeQuery();
85                                  List<Article> result = new ArrayList<>();
86                                  while (rs.next()) {
87                                            result.add(convertArticle(rs));
88                                  }
89                                  return result;
90                        } finally {
91                                  JdbcUtil.close(rs);
92                                  JdbcUtil.close(pstmt);
93                        }
94              }
95
96              private Article convertArticle(ResultSet rs) throws SQLException {
97                        return new Article(rs.getInt("article_no"),
98                                            new Writer(
99                                                                rs.getString("writer_id"),
100                                                               rs.getString("writer_name")),
101                                           rs.getString("title"),
102                                           toDate(rs.getTimestamp("regdate")),
103                                           toDate(rs.getTimestamp("moddate")),
104                                           rs.getInt("read_cnt"));
105             }
106
107             private Date toDate(Timestamp timestamp) {
108                       return new Date(timestamp.getTime());
109             }
110
111             public Article selectById(Connection conn, int no) throws SQLException {
112                       PreparedStatement pstmt = null;
113                       ResultSet rs = null;
114                       try {
115                                 pstmt = conn.prepareStatement(
116                                                     "select * from article where article_no = ?");
117                                 pstmt.setInt(1, no);
118                                 rs = pstmt.executeQuery();
119                                 Article article = null;
120                                 if (rs.next()) {
121                                           article = convertArticle(rs);
122                                 }
```

```
123                                 return article;
124                         } finally {
125                                 JdbcUtil.close(rs);
126                                 JdbcUtil.close(pstmt);
127                         }
128             }
129
130             public void increaseReadCount(Connection conn, int no) throws SQLException {
131                         try (PreparedStatement pstmt =
132                                                 conn.prepareStatement(
133                                                                         "update article set read_cnt = read_cnt + 1
134     "+
135                                                                         "where article_no = ?")) {
136                                 pstmt.setInt(1, no);
137                                 pstmt.executeUpdate();
138                         }
139             }
140
141             public int update(Connection conn, int no, String title) throws SQLException {
142                         try (PreparedStatement pstmt =
143                                                 conn.prepareStatement(
144                                                                         "update article set title = ?, moddate = now()
145     "+
146                                                                         "where article_no = ?")) {
147                                 pstmt.setString(1, title);
148                                 pstmt.setInt(2, no);
149                                 return pstmt.executeUpdate();
150                         }
151             }
152     }
```

(2) ArticleContentDao.java

[board/src/article/dao/ArticleContentDao.java]

```
01    package article.dao;
02
03    import java.sql.Connection;
04    import java.sql.PreparedStatement;
05    import java.sql.ResultSet;
06    import java.sql.SQLException;
07
08    import article.model.ArticleContent;
09    import jdbc.JdbcUtil;
10
11    public class ArticleContentDao {
12
13              public ArticleContent insert(Connection conn, ArticleContent content)
14              throws SQLException {
15                        PreparedStatement pstmt = null;
16                        try {
17                                  pstmt = conn.prepareStatement(
18                                                      "insert into article_content " +
19                                                      "(article_no, content) values (?,?)");
20                                  pstmt.setLong(1, content.getNumber());
21                                  pstmt.setString(2, content.getContent());
22                                  int insertedCount = pstmt.executeUpdate();
23                                  if (insertedCount > 0) {
24                                            return content;
25                                  } else {
26                                            return null;
27                                  }
```

```
28                      } finally {
29                              JdbcUtil.close(pstmt);
30                      }
31              }
32
33              public ArticleContent selectById(Connection conn, int no) throws SQLException {
34                      PreparedStatement pstmt = null;
35                      ResultSet rs = null;
36                      try {
37                              pstmt = conn.prepareStatement(
38                                              "select * from article_content where article_no = ?");
39                              pstmt.setInt(1, no);
40                              rs = pstmt.executeQuery();
41                              ArticleContent content = null;
42                              if (rs.next()) {
43                                      content = new ArticleContent(
44                                                      rs.getInt("article_no"),
45   rs.getString("content"));
46                              }
47                              return content;
48                      } finally {
49                              JdbcUtil.close(rs);
50                              JdbcUtil.close(pstmt);
51                      }
52              }
53
54              public int update(Connection conn, int no, String content) throws SQLException {
55                      try (PreparedStatement pstmt =
56                                      conn.prepareStatement(
57                                                      "update article_content set content = ? "+
58                                                      "where article_no = ?")) {
59                              pstmt.setString(1, content);
60                              pstmt.setInt(2, no);
61                              return pstmt.executeUpdate();
62                      }
63              }
64   }
```

22.4.2 WriteArticleService 구현

(1) WriteRequest.java

[board/src/article/service/WriteRequest.java]

```
01   package article.service;
02
03   import java.util.Map;
04
05   import article.model.Writer;
06
07   public class WriteRequest {
08
09              private Writer writer;
10              private String title;
11              private String content;
12
13              public WriteRequest(Writer writer, String title, String content) {
14                      this.writer = writer;
15                      this.title = title;
16                      this.content = content;
17              }
18
```

```
19              public Writer getWriter() {
20                      return writer;
21              }
22
23              public String getTitle() {
24                      return title;
25              }
26
27              public String getContent() {
28                      return content;
29              }
30
31              public void validate(Map<String, Boolean> errors) {
32                      if (title == null || title.trim().isEmpty()) {
33                              errors.put("title", Boolean.TRUE);
34                      }
35              }
36      }
```

(2) WriteArticleService.java

[board/src/article/service/WriteArticleService.java]

```
01      package article.service;
02
03      import java.sql.Connection;
04      import java.sql.SQLException;
05      import java.util.Date;
06
07      import article.dao.ArticleContentDao;
08      import article.dao.ArticleDao;
09      import article.model.Article;
10      import article.model.ArticleContent;
11      import jdbc.JdbcUtil;
12      import jdbc.connection.ConnectionProvider;
13
14      public class WriteArticleService {
15
16              private ArticleDao articleDao = new ArticleDao();
17              private ArticleContentDao contentDao = new ArticleContentDao();
18
19              public Integer write(WriteRequest req) {
20                      Connection conn = null;
21                      try {
22                              conn = ConnectionProvider.getConnection();
23                              conn.setAutoCommit(false);
24
25                              Article article = toArticle(req);
26                              Article savedArticle = articleDao.insert(conn, article);
27                              if (savedArticle == null) {
28                                      throw new RuntimeException("fail to insert article");
29                              }
30                              ArticleContent content = new ArticleContent(
31                                              savedArticle.getNumber(),
32                                              req.getContent());
33                              ArticleContent savedContent = contentDao.insert(conn, content);
34                              if (savedContent == null) {
35                                      throw new RuntimeException("fail to insert article_content");
36                              }
37
38                              conn.commit();
39
```

```
40                                  return savedArticle.getNumber();
41                          } catch (SQLException e) {
42                                  JdbcUtil.rollback(conn);
43                                  throw new RuntimeException(e);
44                          } catch (RuntimeException e) {
45                                  JdbcUtil.rollback(conn);
46                                  throw e;
47                          } finally {
48                                  JdbcUtil.close(conn);
49                          }
50                  }
51
52                  private Article toArticle(WriteRequest req) {
53                          Date now = new Date();
54                          return new Article(null, req.getWriter(), req.getTitle(), now, now, 0);
55                  }
56          }
```

22.4.3 WriteArticleHandler 구현

(1) WriteArticleHandler

[board/src/article/command/WriteArticleHandler.java]

```
01          package article.command;
02
03          import java.util.HashMap;
04          import java.util.Map;
05
06          import javax.servlet.http.HttpServletRequest;
07          import javax.servlet.http.HttpServletResponse;
08
09          import article.model.Writer;
10          import article.service.WriteArticleService;
11          import article.service.WriteRequest;
12          import auth.service.User;
13          import mvc.command.CommandHandler;
14
15          public class WriteArticleHandler implements CommandHandler {
16                    private static final String FORM_VIEW = "/WEB-INF/view/newArticleForm.jsp";
17                    private WriteArticleService writeService = new WriteArticleService();
18
19                    @Override
20                    public String process(HttpServletRequest req, HttpServletResponse res) {
21                              if (req.getMethod().equalsIgnoreCase("GET")) {
22                                        return processForm(req, res);
23                              } else if (req.getMethod().equalsIgnoreCase("POST")) {
24                                        return processSubmit(req, res);
25                              } else {
26                                        res.setStatus(HttpServletResponse.SC_METHOD_NOT_ALLOWED);
27                                        return null;
28                              }
29                    }
30
31                    private String processForm(HttpServletRequest req, HttpServletResponse res) {
32                              return FORM_VIEW;
33                    }
34
35                    private String processSubmit(HttpServletRequest req, HttpServletResponse res) {
36                              Map<String, Boolean> errors = new HashMap<>();
37                              req.setAttribute("errors", errors);
38
```

```
39                      User user = (User)req.getSession(false).getAttribute("authUser");
40                      WriteRequest writeReq = createWriteRequest(user, req);
41                      writeReq.validate(errors);
42
43                      if (!errors.isEmpty()) {
44                              return FORM_VIEW;
45                      }
46
47                      int newArticleNo = writeService.write(writeReq);
48                      req.setAttribute("newArticleNo", newArticleNo);
49
50                      return "/WEB-INF/view/newArticleSuccess.jsp";
51              }
52
53              private WriteRequest createWriteRequest(User user, HttpServletRequest req) {
54                      return new WriteRequest(
55                                      new Writer(user.getId(), user.getName()),
56                                      req.getParameter("title"),
57                                      req.getParameter("content"));
58              }
59      }
```

(2) commandHandlerURI.properties

■ 핸들러를 사용할 수 있도록 commandHandlerURI.properties 파일에 매핑을 추가한다.

[board/WebContent/WEB-INF/commandHandlerURI.properties]

```
01  /join.do=member.command.JoinHandler
02  /login.do=auth.command.LoginHandler
03  /logout.do=auth.command.LogoutHandler
04  /changePwd.do=member.command.ChangePasswordHandler
05  /article/write.do=article.command.WriteArticleHandler
06  /article/list.do=article.command.ListArticleHandler
07  /article/read.do=article.command.ReadArticleHandler
08  /article/modify.do=article.command.ModifyArticleHandler
```

22.4.4 newArticleForm.jsp와 newArticleSuccess.jsp 구현

(1) newArticleForm.jsp

[board/WebContent/WEB-INF/view/newArticleForm.jsp]

```
01  <%@ page contentType="text/html; charset=utf-8"%>
02  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
03  <!DOCTYPE html>
04  <html>
05  <head>
06  <title>게시글 쓰기</title>
07  </head>
08  <body>
09  <form action="write.do" method="post">
10  <p>
11              제목:<br/><input type="text" name="title" value="${param.title}">
12              <c:if test="${errors.title}">제목을 입력하세요.</c:if>
13  </p>
14  <p>
15              내용:<br/>
```

```
16              <textarea name="content" rows="5" cols="30">${param.title}</textarea>
17      </p>
18      <input type="submit" value="새 글 등록">
19      </form>
20      </body>
21      </html>
```

22.4.5 로그인 검사 필터 적용

(1) web.xml (로그인 검사 필터)

■ 게시글 쓰기 기능은 로그인 한 사용자만 실행할 수 있도록 LoginCheckFilter 매핑 설정을 한다.

[board/WebContent/WEB-INF/web.xml]

```
01      ...(생략)
02
03              <filter>
04                      <filter-name>LoginCheckFilter</filter-name>
05                      <filter-class>filter.LoginCheckFilter</filter-class>
06              </filter>
07              <filter-mapping>
08                      <filter-name>LoginCheckFilter</filter-name>
09                      <url-pattern>/changePwd.do</url-pattern>
10                      <url-pattern>/article/write.do</url-pattern>
11                      <url-pattern>/article/modify.do</url-pattern>
12              </filter-mapping>
13      </web-app>
```

22.4.6 게시글 쓰기 기능 테스트

■ http://localhost:8080/board/article/write.do

22.5 게시판 목록 조회 기능

(1) 게시글 조회 기능의 요구사항

■ 게시글 목록을 요청하면 10개 게시글의 번호, 제목, 작성자, 조회수를 표 형식으로 출력한다.

■ 게시글 목록을 요청할 때 페이지 번호를 지정하지 않으면, 가장 최근에 작성한 게시글 10개를 최근 순으로 보여준다.

(2) 기능을 구현하기 위해 만들 코드 구성

```
ListArticleHandler      → ListArticleService   → ArticleDao        → board DB
        ↓                   ArticlePage
listArticle.jsp
```

22.5.1 ArticleDao의 목록 조회 관련 기능 구현

(1) ArticleDao.java

■ 게시글 목록을 표현하려면 다음의 두 데이터를 가져올 수 있어야 한다.
- 페이지 개수를 구하기 위한 전체 게시글 개수
- 지정한 행 번호에 해당하는 게시글 목록

[board/src/article/dao/ArticleDao.java]

```
01  package article.dao;
02
03  import java.sql.Connection;
04  import java.sql.PreparedStatement;
05  import java.sql.ResultSet;
06  import java.sql.SQLException;
07  import java.sql.Statement;
08  import java.sql.Timestamp;
09  import java.util.ArrayList;
10  import java.util.Date;
11  import java.util.List;
12
13  import article.model.Article;
14  import article.model.Writer;
15  import jdbc.JdbcUtil;
16
17  public class ArticleDao {
18
19          public Article insert(Connection conn, Article article) throws SQLException {
20                  PreparedStatement pstmt = null;
21                  Statement stmt = null;
22                  ResultSet rs = null;
23                  try {
24                          pstmt = conn.prepareStatement("insert into article "
25                                          + "(writer_id, writer_name, title, regdate, moddate,
26  read_cnt) "
27                                          + "values (?,?,?,?,?,0)");
28                          pstmt.setString(1, article.getWriter().getId());
29                          pstmt.setString(2, article.getWriter().getName());
30                          pstmt.setString(3, article.getTitle());
31                          pstmt.setTimestamp(4, toTimestamp(article.getRegDate()));
32                          pstmt.setTimestamp(5, toTimestamp(article.getModifiedDate()));
33                          int insertedCount = pstmt.executeUpdate();
34
35                          if (insertedCount > 0) {
36                                  stmt = conn.createStatement();
37                                  rs = stmt.executeQuery("select last_insert_id() from article");
38                                  if (rs.next()) {
39                                          Integer newNo = rs.getInt(1);
40                                          return new Article(newNo,
41                                                          article.getWriter(),
42                                                          article.getTitle(),
43                                                          article.getRegDate(),
44                                                          article.getModifiedDate(),
45                                                          0);
46                                  }
47                          }
48                          return null;
49                  } finally {
50                          JdbcUtil.close(rs);
51                          JdbcUtil.close(stmt);
52                          JdbcUtil.close(pstmt);
```

```
53                  }
54          }
55
56          private Timestamp toTimestamp(Date date) {
57                  return new Timestamp(date.getTime());
58          }
59
60          public int selectCount(Connection conn) throws SQLException {
61                  Statement stmt = null;
62                  ResultSet rs = null;
63                  try {
64                          stmt = conn.createStatement();
65                          rs = stmt.executeQuery("select count(*) from article");
66                          if (rs.next()) {
67                                  return rs.getInt(1);
68                          }
69                          return 0;
70                  } finally {
71                          JdbcUtil.close(rs);
72                          JdbcUtil.close(stmt);
73                  }
74          }
75
76          public List<Article> select(Connection conn, int startRow, int size) throws SQLException {
77                  PreparedStatement pstmt = null;
78                  ResultSet rs = null;
79                  try {
80                          pstmt = conn.prepareStatement("select * from article " +
81                                          "order by article_no desc limit ?, ?");
82                          pstmt.setInt(1, startRow);
83                          pstmt.setInt(2, size);
84                          rs = pstmt.executeQuery();
85                          List<Article> result = new ArrayList<>();
86                          while (rs.next()) {
87                                  result.add(convertArticle(rs));
88                          }
89                          return result;
90                  } finally {
91                          JdbcUtil.close(rs);
92                          JdbcUtil.close(pstmt);
93                  }
94          }
95
96          private Article convertArticle(ResultSet rs) throws SQLException {
97                  return new Article(rs.getInt("article_no"),
98                                  new Writer(
99                                                  rs.getString("writer_id"),
100                                                 rs.getString("writer_name")),
101                                 rs.getString("title"),
102                                 toDate(rs.getTimestamp("regdate")),
103                                 toDate(rs.getTimestamp("moddate")),
104                                 rs.getInt("read_cnt"));
105         }
106
107         private Date toDate(Timestamp timestamp) {
108                 return new Date(timestamp.getTime());
109         }
110
111         public Article selectById(Connection conn, int no) throws SQLException {
112                 PreparedStatement pstmt = null;
113                 ResultSet rs = null;
114                 try {
115                         pstmt = conn.prepareStatement(
116                                         "select * from article where article_no = ?");
117                         pstmt.setInt(1, no);
```

```
118                                 rs = pstmt.executeQuery();
119                                 Article article = null;
120                                 if (rs.next()) {
121                                         article = convertArticle(rs);
122                                 }
123                                 return article;
124                         } finally {
125                                 JdbcUtil.close(rs);
126                                 JdbcUtil.close(pstmt);
127                         }
128                 }
129
130                 public void increaseReadCount(Connection conn, int no) throws SQLException {
131                         try (PreparedStatement pstmt =
132                                         conn.prepareStatement(
133                                                         "update article set read_cnt = read_cnt + 1
134         "+
135                                                         "where article_no = ?")) {
136                                 pstmt.setInt(1, no);
137                                 pstmt.executeUpdate();
138                         }
139                 }
140
141                 public int update(Connection conn, int no, String title) throws SQLException {
142                         try (PreparedStatement pstmt =
143                                         conn.prepareStatement(
144                                                         "update article set title = ?, moddate = now()
145         "+
146                                                         "where article_no = ?")) {
147                                 pstmt.setString(1, title);
148                                 pstmt.setInt(2, no);
149                                 return pstmt.executeUpdate();
150                         }
151                 }
152         }
```

22.5.2 ArticlePage와 ListArticleService 구현

(1) ArticlePage.java

(2) ListArticleService.java

22.5.3 ListArticleHandler 구현

(1) ListArticleHandler.java

[board/src/article/command/ListArticleHandler.java]

```
01    package article.command;
02
03    import javax.servlet.http.HttpServletRequest;
04    import javax.servlet.http.HttpServletResponse;
05
06    import article.service.ArticlePage;
07    import article.service.ListArticleService;
08    import mvc.command.CommandHandler;
09
10    public class ListArticleHandler implements CommandHandler {
11
```

```
12              private ListArticleService listService = new ListArticleService();
13
14              @Override
15              public String process(HttpServletRequest req, HttpServletResponse res)
16                                      throws Exception {
17                      String pageNoVal = req.getParameter("pageNo");
18                      int pageNo = 1;
19                      if (pageNoVal != null) {
20                              pageNo = Integer.parseInt(pageNoVal);
21                      }
22                      ArticlePage articlePage = listService.getArticlePage(pageNo);
23                      req.setAttribute("articlePage", articlePage);
24                      return "/WEB-INF/view/listArticle.jsp";
25              }
26
27      }
```

(2) commandHandlerURI.properties

■ 핸들러를 사용할 수 있도록 commandHandlerURI.properties 파일에 매핑을 추가한다.

[board/WebContent/WEB-INF/commandHandlerURI.properties]

```
01      /join.do=member.command.JoinHandler
02      /login.do=auth.command.LoginHandler
03      /logout.do=auth.command.LogoutHandler
04      /changePwd.do=member.command.ChangePasswordHandler
05      /article/write.do=article.command.WriteArticleHandler
06      /article/list.do=article.command.ListArticleHandler
07      /article/read.do=article.command.ReadArticleHandler
08      /article/modify.do=article.command.ModifyArticleHandler
```

22.5.4 listArticle.jsp 구현

(1) listArticle.jsp

[board/src/article/command/WriteArticleHandler.java]

```
01      <%@ page contentType="text/html; charset=utf-8"%>
02      <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
03      <!DOCTYPE html>
04      <html>
05      <head>
06      <title>게시글 목록</title>
07      </head>
08      <body>
09
10      <table border="1">
11              <tr>
12                      <td colspan="4"><a href="write.do">[게시글쓰기]</a></td>
13              </tr>
14              <tr>
15                      <td>번호</td>
16                      <td>제목</td>
17                      <td>작성자</td>
18                      <td>조회수</td>
19              </tr>
20      <c:if test="${articlePage.hasNoArticles()}">
```

```
21              <tr>
22                      <td colspan="4">게시글이 없습니다.</td>
23              </tr>
24      </c:if>
25      <c:forEach var="article" items="${articlePage.content}">
26              <tr>
27                      <td>${article.number}</td>
28                      <td>
29                      <a href="read.do?no=${article.number}&pageNo=${articlePage.currentPage}">
30                      <c:out value="${article.title}"/>
31                      </a>
32                      </td>
33                      <td>${article.writer.name}</td>
34                      <td>${article.readCount}</td>
35              </tr>
36      </c:forEach>
37      <c:if test="${articlePage.hasArticles()}">
38              <tr>
39                      <td colspan="4">
40                              <c:if test="${articlePage.startPage > 5}">
41                              <a href="list.do?pageNo=${articlePage.startPage - 5}">[이전]</a>
42                              </c:if>
43                              <c:forEach var="pNo"
44                                                      begin="${articlePage.startPage}"
45                                                      end="${articlePage.endPage}">
46                              <a href="list.do?pageNo=${pNo}">[${pNo}]</a>
47                              </c:forEach>
48                              <c:if test="${articlePage.endPage < articlePage.totalPages}">
49                              <a href="list.do?pageNo=${articlePage.startPage + 5}">[다음]</a>
50                              </c:if>
51                      </td>
52              </tr>
53      </c:if>
54      </table>
55      </body>
56      </html>
```

22.5.5 게시글 목록 조회 기능 테스트

- http://localhost:8080/board/article/list.do

22.6 게시글 조회 기능

(1) 게시글 조회 기능의 요구사항

- 요청한 번호에 해당하는 게시글의 내용을 출력한다.
- 게시글 조회수를 1 증가한다.
- 존재하지 않는 게시글 번호를 요청하면 페이지 없음 상태 코드(404)를 응답한다.

(2) 기능을 구현하기 위해 만들 코드 구성

```
ReadArticleHandler      → ReadArticleService,  → ArticleDao,          → board DB
        ↓                   AricleData             ArticleContentDao
```

readArticle.jsp

22.6.1 ArticleDao의 조회 관련 기능 구현

(1) ArticleDao.java

[board/src/article/dao/ArticleDao.java]

```
01    package article.dao;
02
03    import java.sql.Connection;
04    import java.sql.PreparedStatement;
05    import java.sql.ResultSet;
06    import java.sql.SQLException;
07    import java.sql.Statement;
08    import java.sql.Timestamp;
09    import java.util.ArrayList;
10    import java.util.Date;
11    import java.util.List;
12
13    import article.model.Article;
14    import article.model.Writer;
15    import jdbc.JdbcUtil;
16
17    public class ArticleDao {
18
19              public Article insert(Connection conn, Article article) throws SQLException {
20                        PreparedStatement pstmt = null;
21                        Statement stmt = null;
22                        ResultSet rs = null;
23                        try {
24                                  pstmt = conn.prepareStatement("insert into article "
25                                                      + "(writer_id, writer_name, title, regdate, moddate,
26    read_cnt) "
27                                                      + "values (?,?,?,?,?,0)");
28                                  pstmt.setString(1, article.getWriter().getId());
29                                  pstmt.setString(2, article.getWriter().getName());
30                                  pstmt.setString(3, article.getTitle());
31                                  pstmt.setTimestamp(4, toTimestamp(article.getRegDate()));
32                                  pstmt.setTimestamp(5, toTimestamp(article.getModifiedDate()));
33                                  int insertedCount = pstmt.executeUpdate();
34
35                                  if (insertedCount > 0) {
36                                            stmt = conn.createStatement();
37                                            rs = stmt.executeQuery("select last_insert_id() from article");
38                                            if (rs.next()) {
39                                                      Integer newNo = rs.getInt(1);
40                                                      return new Article(newNo,
41                                                                          article.getWriter(),
42                                                                          article.getTitle(),
43                                                                          article.getRegDate(),
44                                                                          article.getModifiedDate(),
45                                                                          0);
46                                            }
47                                  }
48                                  return null;
49                        } finally {
50                                  JdbcUtil.close(rs);
51                                  JdbcUtil.close(stmt);
52                                  JdbcUtil.close(pstmt);
53                        }
54              }
```

```
55
56              private Timestamp toTimestamp(Date date) {
57                      return new Timestamp(date.getTime());
58              }
59
60              public int selectCount(Connection conn) throws SQLException {
61                      Statement stmt = null;
62                      ResultSet rs = null;
63                      try {
64                              stmt = conn.createStatement();
65                              rs = stmt.executeQuery("select count(*) from article");
66                              if (rs.next()) {
67                                      return rs.getInt(1);
68                              }
69                              return 0;
70                      } finally {
71                              JdbcUtil.close(rs);
72                              JdbcUtil.close(stmt);
73                      }
74              }
75
76              public List<Article> select(Connection conn, int startRow, int size) throws SQLException {
77                      PreparedStatement pstmt = null;
78                      ResultSet rs = null;
79                      try {
80                              pstmt = conn.prepareStatement("select * from article " +
81                                              "order by article_no desc limit ?, ?");
82                              pstmt.setInt(1, startRow);
83                              pstmt.setInt(2, size);
84                              rs = pstmt.executeQuery();
85                              List<Article> result = new ArrayList<>();
86                              while (rs.next()) {
87                                      result.add(convertArticle(rs));
88                              }
89                              return result;
90                      } finally {
91                              JdbcUtil.close(rs);
92                              JdbcUtil.close(pstmt);
93                      }
94              }
95
96              private Article convertArticle(ResultSet rs) throws SQLException {
97                      return new Article(rs.getInt("article_no"),
98                                      new Writer(
99                                                      rs.getString("writer_id"),
100                                                     rs.getString("writer_name")),
101                                     rs.getString("title"),
102                                     toDate(rs.getTimestamp("regdate")),
103                                     toDate(rs.getTimestamp("moddate")),
104                                     rs.getInt("read_cnt"));
105             }
106
107             private Date toDate(Timestamp timestamp) {
108                     return new Date(timestamp.getTime());
109             }
110
111             public Article selectById(Connection conn, int no) throws SQLException {
112                     PreparedStatement pstmt = null;
113                     ResultSet rs = null;
114                     try {
115                             pstmt = conn.prepareStatement(
116                                             "select * from article where article_no = ?");
117                             pstmt.setInt(1, no);
118                             rs = pstmt.executeQuery();
119                             Article article = null;
```

```
120                             if (rs.next()) {
121                                     article = convertArticle(rs);
122                             }
123                             return article;
124                     } finally {
125                             JdbcUtil.close(rs);
126                             JdbcUtil.close(pstmt);
127                     }
128             }
129
130             public void increaseReadCount(Connection conn, int no) throws SQLException {
131                     try (PreparedStatement pstmt =
132                                     conn.prepareStatement(
133                                                     "update article set read_cnt = read_cnt + 1
134 "+
135                                                     "where article_no = ?")) {
136                             pstmt.setInt(1, no);
137                             pstmt.executeUpdate();
138                     }
139             }
140
141             public int update(Connection conn, int no, String title) throws SQLException {
142                     try (PreparedStatement pstmt =
143                                     conn.prepareStatement(
144                                                     "update article set title = ?, moddate = now()
145 "+
146                                                     "where article_no = ?")) {
147                             pstmt.setString(1, title);
148                             pstmt.setInt(2, no);
149                             return pstmt.executeUpdate();
150                     }
151             }
152 }
```

22.6.2 ArticleContentDao의 조회 관련 기능 구현

(1) ArticleContentDao.java

[board/src/article/dao/ArticleContentDao.java]

```
01  package article.dao;
02
03  import java.sql.Connection;
04  import java.sql.PreparedStatement;
05  import java.sql.ResultSet;
06  import java.sql.SQLException;
07
08  import article.model.ArticleContent;
09  import jdbc.JdbcUtil;
10
11  public class ArticleContentDao {
12
13          public ArticleContent insert(Connection conn, ArticleContent content)
14          throws SQLException {
15                  PreparedStatement pstmt = null;
16                  try {
17                          pstmt = conn.prepareStatement(
18                                          "insert into article_content " +
19                                          "(article_no, content) values (?,?)");
20                          pstmt.setLong(1, content.getNumber());
21                          pstmt.setString(2, content.getContent());
22                          int insertedCount = pstmt.executeUpdate();
```

```
23                                  if (insertedCount > 0) {
24                                          return content;
25                                  } else {
26                                          return null;
27                                  }
28                          } finally {
29                                  JdbcUtil.close(pstmt);
30                          }
31                  }
32
33                  public ArticleContent selectById(Connection conn, int no) throws SQLException {
34                          PreparedStatement pstmt = null;
35                          ResultSet rs = null;
36                          try {
37                                  pstmt = conn.prepareStatement(
38                                                  "select * from article_content where article_no = ?");
39                                  pstmt.setInt(1, no);
40                                  rs = pstmt.executeQuery();
41                                  ArticleContent content = null;
42                                  if (rs.next()) {
43                                          content = new ArticleContent(
44                                                          rs.getInt("article_no"),
45          rs.getString("content"));
46                                  }
47                                  return content;
48                          } finally {
49                                  JdbcUtil.close(rs);
50                                  JdbcUtil.close(pstmt);
51                          }
52                  }
53
54                  public int update(Connection conn, int no, String content) throws SQLException {
55                          try (PreparedStatement pstmt =
56                                          conn.prepareStatement(
57                                                          "update article_content set content = ? "+
58                                                          "where article_no = ?")) {
59                                  pstmt.setString(1, content);
60                                  pstmt.setInt(2, no);
61                                  return pstmt.executeUpdate();
62                          }
63                  }
64          }
```

22.6.3 ArticleData 클래스 구현

(1) ArticleData.java

[board/src/article/service/ArticleData.java]

```
01          package article.service;
02
03          import article.model.Article;
04          import article.model.ArticleContent;
05
06          public class ArticleData {
07
08                  private Article article;
09                  private ArticleContent content;
10
11                  public ArticleData(Article article, ArticleContent content) {
12                          this.article = article;
13                          this.content = content;
```

```
14              }
15
16              public Article getArticle() {
17                      return article;
18              }
19
20              public String getContent() {
21                      return content.getContent();
22              }
23
24      }
```

22.6.4 데이터가 없음을 의미하는 익셉션 클래스 구현

(1) ArticleNotFoundException.java

(2) ArticleContentNotFoundException.java

22.6.5 ReadArticleService 구현

(1) ReadArticleService.java

[board/src/article/service/ReadArticleService.java]

```
01      package article.service;
02
03      import java.sql.Connection;
04      import java.sql.SQLException;
05
06      import article.dao.ArticleContentDao;
07      import article.dao.ArticleDao;
08      import article.model.Article;
09      import article.model.ArticleContent;
10      import jdbc.connection.ConnectionProvider;
11
12      public class ReadArticleService {
13
14              private ArticleDao articleDao = new ArticleDao();
15              private ArticleContentDao contentDao = new ArticleContentDao();
16
17              public ArticleData getArticle(int articleNum, boolean increaseReadCount) {
18                      try (Connection conn = ConnectionProvider.getConnection()){
19                              Article article = articleDao.selectById(conn, articleNum);
20                              if (article == null) {
21                                      throw new ArticleNotFoundException();
22                              }
23                              ArticleContent content = contentDao.selectById(conn, articleNum);
24                              if (content == null) {
25                                      throw new ArticleContentNotFoundException();
26                              }
27                              if (increaseReadCount) {
28                                      articleDao.increaseReadCount(conn, articleNum);
29                              }
30                              return new ArticleData(article, content);
31                      } catch (SQLException e) {
32                              throw new RuntimeException(e);
33                      }
34              }
35      }
```

22.6.6 ReadArticleHandler 구현

(1) ReadArticleHandler

[board/src/article/command/ReadArticleHandler.java]

```
01    package article.command;
02
03    import javax.servlet.http.HttpServletRequest;
04    import javax.servlet.http.HttpServletResponse;
05
06    import article.service.ArticleContentNotFoundException;
07    import article.service.ArticleData;
08    import article.service.ArticleNotFoundException;
09    import article.service.ReadArticleService;
10    import mvc.command.CommandHandler;
11
12    public class ReadArticleHandler implements CommandHandler {
13
14              private ReadArticleService readService = new ReadArticleService();
15
16              @Override
17              public String process(HttpServletRequest req, HttpServletResponse res)
18                                        throws Exception {
19                        String noVal = req.getParameter("no");
20                        int articleNum = Integer.parseInt(noVal);
21                        try {
22                                  ArticleData articleData = readService.getArticle(articleNum, true);
23                                  req.setAttribute("articleData", articleData);
24                                  return "/WEB-INF/view/readArticle.jsp";
25                        } catch (ArticleNotFoundException | ArticleContentNotFoundException e) {
26                                  req.getServletContext().log("no article", e);
27                                  res.sendError(HttpServletResponse.SC_NOT_FOUND);
28                                  return null;
29                        }
30              }
31
32    }
```

(2) commandHandlerURI.properties

■ 핸들러를 사용할 수 있도록 commandHandlerURI.properties 파일에 매핑을 추가한다.

[board/WebContent/WEB-INF/commandHandlerURI.properties]

```
01    /join.do=member.command.JoinHandler
02    /login.do=auth.command.LoginHandler
03    /logout.do=auth.command.LogoutHandler
04    /changePwd.do=member.command.ChangePasswordHandler
05    /article/write.do=article.command.WriteArticleHandler
06    /article/list.do=article.command.ListArticleHandler
07    /article/read.do=article.command.ReadArticleHandler
08    /article/modify.do=article.command.ModifyArticleHandler
```

22.6.7 JSP에서 내용 출력을 위한 커스텀 태그 구현

(1) pre.tag

■ 최대한 폼에 입력한 내용을 그대로 보여주기 위해 치환 작업을 해주는 커스텀 태그를 작성한다.

[board/WebContent/WEB-INF/tag/pre.tag]

```
<%@ tag body-content="empty" pageEncoding="utf-8" %>
<%@ tag trimDirectiveWhitespaces="true" %>
<%@ attribute name="value" type="java.lang.String" required="true"%>
<%
          value = value.replace("&", "&amp;");
          value = value.replace("<", "&lt;");
          value = value.replace(" ", " ");
          value = value.replace("\n", "\n<br>");
%>
<%= value %>
```

22.6.8 readArticle.jsp 구현

(1) readArticle.jsp

[board/WebContent/WEB-INF/view/readArticle.jsp]

```
<%@ page contentType="text/html; charset=utf-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="u" tagdir="/WEB-INF/tags" %>
<!DOCTYPE html>
<html>
<head>
<title>게시글 읽기</title>
</head>
<body>
<table border="1" width="100%">
<tr>
          <td>번호</td>
          <td>${articleData.article.number}</td>
</tr>
<tr>
          <td>작성자</td>
          <td>${articleData.article.writer.name}</td>
</tr>
<tr>
          <td>제목</td>
          <td><c:out value='${articleData.article.title}' /></td>
</tr>
<tr>
          <td>내용</td>
          <td><u:pre value='${articleData.content}'/></td>
</tr>
<tr>
          <td colspan="2">
                    <c:set var="pageNo" value="${empty param.pageNo ? '1' : param.pageNo}" />
                    <a href="list.do?pageNo=${pageNo}">[목록]</a>
                    <c:if test="${authUser.id == articleData.article.writer.id}">
                    <a href="modify.do?no=${articleData.article.number}">[게시글수정]</a>
                    <a href="delete.do?no=${articleData.article.number}">[게시글삭제]</a>
                    </c:if>
```

```
35              </td>
36      </tr>
37      </table>
38
39      </body>
40      </html>
```

22.6.9 게시글 조회 기능 테스트

■ 게시글 목록에서 게시글 제목의 링크를 클릭한다.

22.7 게시글 수정 기능

(1) 게시글 수정 기능의 요구사항

■ 게시글 수정 폼은 게시글 제목과 내용을 표시한다.

■ 게시글 수정은 작성자만 할 수 있다.

(2) 기능을 구현하기 위해 만들 코드 구성

```
ModifyArticleHandler    → ModifyArticleService        → ArticleDao          →   board
DB
        ↓                   ModifyRequest                 ArticleContentDao
modifyForm.jsp
modifySuccess.jsp
```

22.7.1 ArticleDao와 ArticleContentDao의 수정 관련 기능 구현

(1) ArticleDao.java

[board/src/article/dao/ArticleDao.java]

```
01    package article.dao;
02
03    import java.sql.Connection;
04    import java.sql.PreparedStatement;
05    import java.sql.ResultSet;
06    import java.sql.SQLException;
07    import java.sql.Statement;
08    import java.sql.Timestamp;
09    import java.util.ArrayList;
10    import java.util.Date;
11    import java.util.List;
12
13    import article.model.Article;
14    import article.model.Writer;
15    import jdbc.JdbcUtil;
16
17    public class ArticleDao {
18
19              public Article insert(Connection conn, Article article) throws SQLException {
```

```
20                      PreparedStatement pstmt = null;
21                      Statement stmt = null;
22                      ResultSet rs = null;
23                      try {
24                                      pstmt = conn.prepareStatement("insert into article "
25                                                      + "(writer_id, writer_name, title, regdate, moddate,
26   read_cnt) "
27                                                      + "values (?,?,?,?,?,0)");
28                                      pstmt.setString(1, article.getWriter().getId());
29                                      pstmt.setString(2, article.getWriter().getName());
30                                      pstmt.setString(3, article.getTitle());
31                                      pstmt.setTimestamp(4, toTimestamp(article.getRegDate()));
32                                      pstmt.setTimestamp(5, toTimestamp(article.getModifiedDate()));
33                                      int insertedCount = pstmt.executeUpdate();
34
35                                      if (insertedCount > 0) {
36                                                      stmt = conn.createStatement();
37                                                      rs = stmt.executeQuery("select last_insert_id() from article");
38                                                      if (rs.next()) {
39                                                                      Integer newNo = rs.getInt(1);
40                                                                      return new Article(newNo,
41                                                                                      article.getWriter(),
42                                                                                      article.getTitle(),
43                                                                                      article.getRegDate(),
44                                                                                      article.getModifiedDate(),
45                                                                                      0);
46                                                      }
47                                      }
48                                      return null;
49                      } finally {
50                                      JdbcUtil.close(rs);
51                                      JdbcUtil.close(stmt);
52                                      JdbcUtil.close(pstmt);
53                      }
54              }
55
56              private Timestamp toTimestamp(Date date) {
57                      return new Timestamp(date.getTime());
58              }
59
60              public int selectCount(Connection conn) throws SQLException {
61                      Statement stmt = null;
62                      ResultSet rs = null;
63                      try {
64                                      stmt = conn.createStatement();
65                                      rs = stmt.executeQuery("select count(*) from article");
66                                      if (rs.next()) {
67                                                      return rs.getInt(1);
68                                      }
69                                      return 0;
70                      } finally {
71                                      JdbcUtil.close(rs);
72                                      JdbcUtil.close(stmt);
73                      }
74              }
75
76              public List<Article> select(Connection conn, int startRow, int size) throws SQLException {
77                      PreparedStatement pstmt = null;
78                      ResultSet rs = null;
79                      try {
80                                      pstmt = conn.prepareStatement("select * from article " +
81                                                      "order by article_no desc limit ?, ?");
82                                      pstmt.setInt(1, startRow);
83                                      pstmt.setInt(2, size);
84                                      rs = pstmt.executeQuery();
```

```
85                                  List<Article> result = new ArrayList<>();
86                                  while (rs.next()) {
87                                          result.add(convertArticle(rs));
88                                  }
89                                  return result;
90                          } finally {
91                                  JdbcUtil.close(rs);
92                                  JdbcUtil.close(pstmt);
93                          }
94                  }
95
96                  private Article convertArticle(ResultSet rs) throws SQLException {
97                          return new Article(rs.getInt("article_no"),
98                                  new Writer(
99                                                  rs.getString("writer_id"),
100                                                 rs.getString("writer_name")),
101                                 rs.getString("title"),
102                                 toDate(rs.getTimestamp("regdate")),
103                                 toDate(rs.getTimestamp("moddate")),
104                                 rs.getInt("read_cnt"));
105                 }
106
107                 private Date toDate(Timestamp timestamp) {
108                         return new Date(timestamp.getTime());
109                 }
110
111                 public Article selectById(Connection conn, int no) throws SQLException {
112                         PreparedStatement pstmt = null;
113                         ResultSet rs = null;
114                         try {
115                                 pstmt = conn.prepareStatement(
116                                                 "select * from article where article_no = ?");
117                                 pstmt.setInt(1, no);
118                                 rs = pstmt.executeQuery();
119                                 Article article = null;
120                                 if (rs.next()) {
121                                         article = convertArticle(rs);
122                                 }
123                                 return article;
124                         } finally {
125                                 JdbcUtil.close(rs);
126                                 JdbcUtil.close(pstmt);
127                         }
128                 }
129
130                 public void increaseReadCount(Connection conn, int no) throws SQLException {
131                         try (PreparedStatement pstmt =
132                                         conn.prepareStatement(
133                                                         "update article set read_cnt = read_cnt + 1
134 "+
135                                                         "where article_no = ?")) {
136                                 pstmt.setInt(1, no);
137                                 pstmt.executeUpdate();
138                         }
139                 }
140
141                 public int update(Connection conn, int no, String title) throws SQLException {
142                         try (PreparedStatement pstmt =
143                                         conn.prepareStatement(
144                                                         "update article set title = ?, moddate = now()
145 "+
146                                                         "where article_no = ?")) {
147                                 pstmt.setString(1, title);
148                                 pstmt.setInt(2, no);
149                                 return pstmt.executeUpdate();
```

```
150                     }
151             }
152     }
```

(2) ArticleContentDao.java

[board/src/article/dao/ArticleContentDao.java]

```
01    package article.dao;
02
03    import java.sql.Connection;
04    import java.sql.PreparedStatement;
05    import java.sql.ResultSet;
06    import java.sql.SQLException;
07
08    import article.model.ArticleContent;
09    import jdbc.JdbcUtil;
10
11    public class ArticleContentDao {
12
13              public ArticleContent insert(Connection conn, ArticleContent content)
14              throws SQLException {
15                        PreparedStatement pstmt = null;
16                        try {
17                                  pstmt = conn.prepareStatement(
18                                                      "insert into article_content " +
19                                                      "(article_no, content) values (?,?)");
20                                  pstmt.setLong(1, content.getNumber());
21                                  pstmt.setString(2, content.getContent());
22                                  int insertedCount = pstmt.executeUpdate();
23                                  if (insertedCount > 0) {
24                                            return content;
25                                  } else {
26                                            return null;
27                                  }
28                        } finally {
29                                  JdbcUtil.close(pstmt);
30                        }
31              }
32
33              public ArticleContent selectById(Connection conn, int no) throws SQLException {
34                        PreparedStatement pstmt = null;
35                        ResultSet rs = null;
36                        try {
37                                  pstmt = conn.prepareStatement(
38                                                      "select * from article_content where article_no = ?");
39                                  pstmt.setInt(1, no);
40                                  rs = pstmt.executeQuery();
41                                  ArticleContent content = null;
42                                  if (rs.next()) {
43                                            content = new ArticleContent(
44                                                                rs.getInt("article_no"),
45    rs.getString("content"));
46                                  }
47                                  return content;
48                        } finally {
49                                  JdbcUtil.close(rs);
50                                  JdbcUtil.close(pstmt);
51                        }
52              }
53
54              public int update(Connection conn, int no, String content) throws SQLException {
```

```
55                      try (PreparedStatement pstmt =
56                                      conn.prepareStatement(
57                                                      "update article_content set content = ? "+
58                                                      "where article_no = ?")) {
59                              pstmt.setString(1, content);
60                              pstmt.setInt(2, no);
61                              return pstmt.executeUpdate();
62                      }
63              }
64      }
```

22.7.2 게시글을 수정할 수 없을 때 사용할 익셉션 구현

(1) PermissionDeniedException.java

22.7.3 ModifyRequest와 ModifyArticleService 구현

(1) ModifyRequest.java

■ 게시글을 수정하려면 수정할 게시글 번호, 수정하는 사용자 아이디, 수정할 제목, 수정할 내용 데이터가 필요하다. 이 데이터를 담기 위한 ModifyRequest 클래스를 작성한다.

[board/src/article/service/ModifyRequest.java]

```
01  package article.service;
02
03  import java.util.Map;
04
05  public class ModifyRequest {
06
07              private String userId;
08              private int articleNumber;
09              private String title;
10              private String content;
11
12              public ModifyRequest(String userId, int articleNumber, String title, String content) {
13                      this.userId = userId;
14                      this.articleNumber = articleNumber;
15                      this.title = title;
16                      this.content = content;
17              }
18
19              public String getUserId() {
20                      return userId;
21              }
22
23              public int getArticleNumber() {
24                      return articleNumber;
25              }
26
27              public String getTitle() {
28                      return title;
29              }
30
31              public String getContent() {
32                      return content;
33              }
34
```

```
35              public void validate(Map<String, Boolean> errors) {
36                      if (title == null || title.trim().isEmpty()) {
37                              errors.put("title", Boolean.TRUE);
38                      }
39              }
40
41      }
```

(2) ModifyArticleService.java

■ ModifyRequest를 이용해서 게시글 수정 기능을 제공한다.

[board/src/article/service/ModifyArticleService.java]

```
01      package article.service;
02
03      import java.sql.Connection;
04      import java.sql.SQLException;
05
06      import article.dao.ArticleContentDao;
07      import article.dao.ArticleDao;
08      import article.model.Article;
09      import jdbc.JdbcUtil;
10      import jdbc.connection.ConnectionProvider;
11
12      public class ModifyArticleService {
13
14              private ArticleDao articleDao = new ArticleDao();
15              private ArticleContentDao contentDao = new ArticleContentDao();
16
17              public void modify(ModifyRequest modReq) {
18                      Connection conn = null;
19                      try {
20                              conn = ConnectionProvider.getConnection();
21                              conn.setAutoCommit(false);
22
23                              Article article = articleDao.selectById(conn,
24                                              modReq.getArticleNumber());
25                              if (article == null) {
26                                      throw new ArticleNotFoundException();
27                              }
28                              if (!canModify(modReq.getUserId(), article)) {
29                                      throw new PermissionDeniedException();
30                              }
31                              articleDao.update(conn,
32                                              modReq.getArticleNumber(), modReq.getTitle());
33                              contentDao.update(conn,
34                                              modReq.getArticleNumber(), modReq.getContent());
35                              conn.commit();
36                      } catch (SQLException e) {
37                              JdbcUtil.rollback(conn);
38                              throw new RuntimeException(e);
39                      } catch (PermissionDeniedException e) {
40                              JdbcUtil.rollback(conn);
41                              throw e;
42                      } finally {
43                              JdbcUtil.close(conn);
44                      }
45              }
46
47              private boolean canModify(String modfyingUserId, Article article) {
48                      return article.getWriter().getId().equals(modfyingUserId);
```

```
49            }
50    }
```

22.7.4 ModifyArticleHandler 구현

(1) ModifyArticleHandler.java

[board/src/article/command/WriteArticleHandler.java]

```
01    package article.command;
02
03    import java.io.IOException;
04    import java.util.HashMap;
05    import java.util.Map;
06
07    import javax.servlet.http.HttpServletRequest;
08    import javax.servlet.http.HttpServletResponse;
09
10    import article.service.ArticleData;
11    import article.service.ArticleNotFoundException;
12    import article.service.ModifyArticleService;
13    import article.service.ModifyRequest;
14    import article.service.PermissionDeniedException;
15    import article.service.ReadArticleService;
16    import auth.service.User;
17    import mvc.command.CommandHandler;
18
19    public class ModifyArticleHandler implements CommandHandler {
20              private static final String FORM_VIEW = "/WEB-INF/view/modifyForm.jsp";
21
22              private ReadArticleService readService = new ReadArticleService();
23              private ModifyArticleService modifyService = new ModifyArticleService();
24
25              @Override
26              public String process(HttpServletRequest req, HttpServletResponse res)
27                                throws Exception {
28                        if (req.getMethod().equalsIgnoreCase("GET")) {
29                                  return processForm(req, res);
30                        } else if (req.getMethod().equalsIgnoreCase("POST")) {
31                                  return processSubmit(req, res);
32                        } else {
33                                  res.setStatus(HttpServletResponse.SC_METHOD_NOT_ALLOWED);
34                                  return null;
35                        }
36              }
37
38              private String processForm(HttpServletRequest req, HttpServletResponse res)
39                                throws IOException {
40                        try {
41                                  String noVal = req.getParameter("no");
42                                  int no = Integer.parseInt(noVal);
43                                  ArticleData articleData = readService.getArticle(no, false);
44                                  User authUser = (User) req.getSession().getAttribute("authUser");
45                                  if (!canModify(authUser, articleData)) {
46                                            res.sendError(HttpServletResponse.SC_FORBIDDEN);
47                                            return null;
48                                  }
49                                  ModifyRequest modReq = new ModifyRequest(authUser.getId(), no,
50                                                      articleData.getArticle().getTitle(),
51                                                      articleData.getContent());
52
53                                  req.setAttribute("modReq", modReq);
```

```
54                                  return FORM_VIEW;
55                          } catch (ArticleNotFoundException e) {
56                                  res.sendError(HttpServletResponse.SC_NOT_FOUND);
57                                  return null;
58                          }
59                  }
60
61                  private boolean canModify(User authUser, ArticleData articleData) {
62                          String writerId = articleData.getArticle().getWriter().getId();
63                          return authUser.getId().equals(writerId);
64                  }
65
66                  private String processSubmit(HttpServletRequest req, HttpServletResponse res)
67                                  throws Exception {
68                          User authUser = (User) req.getSession().getAttribute("authUser");
69                          String noVal = req.getParameter("no");
70                          int no = Integer.parseInt(noVal);
71
72                          ModifyRequest modReq = new ModifyRequest(authUser.getId(), no,
73                                          req.getParameter("title"),
74                                          req.getParameter("content"));
75                          req.setAttribute("modReq", modReq);
76
77                          Map<String, Boolean> errors = new HashMap<>();
78                          req.setAttribute("errors", errors);
79                          modReq.validate(errors);
80                          if (!errors.isEmpty()) {
81                                  return FORM_VIEW;
82                          }
83                          try {
84                                  modifyService.modify(modReq);
85                                  return "/WEB-INF/view/modifySuccess.jsp";
86                          } catch (ArticleNotFoundException e) {
87                                  res.sendError(HttpServletResponse.SC_NOT_FOUND);
88                                  return null;
89                          } catch (PermissionDeniedException e) {
90                                  res.sendError(HttpServletResponse.SC_FORBIDDEN);
91                                  return null;
92                          }
93                  }
94          }
```

(2) commandHandlerURI.properties

■ 핸들러를 사용할 수 있도록 commandHandlerURI.properties 파일에 매핑을 추가한다.

[board/WebContent/WEB-INF/commandHandlerURI.properties]

```
01  /join.do=member.command.JoinHandler
02  /login.do=auth.command.LoginHandler
03  /logout.do=auth.command.LogoutHandler
04  /changePwd.do=member.command.ChangePasswordHandler
05  /article/write.do=article.command.WriteArticleHandler
06  /article/list.do=article.command.ListArticleHandler
07  /article/read.do=article.command.ReadArticleHandler
08  /article/modify.do=article.command.ModifyArticleHandler
```

22.7.5 modifyForm.jsp와 modifySuccess.jsp 구현

(1) modifyForm.jsp

[board/WebContent/WEB-INF/view/modifyForm.jsp]

```
01  <%@ page contentType="text/html; charset=utf-8"%>
02  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
03  <!DOCTYPE html>
04  <html>
05  <head>
06  <title>게시글 수정</title>
07  </head>
08  <body>
09  <form action="modify.do" method="post">
10  <input type="hidden" name="no" value="${modReq.articleNumber}">
11  <p>
12            번호:<br/>${modReq.articleNumber}
13  </p>
14  <p>
15            제목:<br/><input type="text" name="title" value="${modReq.title}">
16            <c:if test="${errors.title}">제목을 입력하세요.</c:if>
17  </p>
18  <p>
19            내용:<br/>
20            <textarea name="content" rows="5" cols="30">${modReq.content}</textarea>
21  </p>
22  <input type="submit" value="글 수정">
23  </form>
24  </body>
25  </html>
```

(2) modifySuccess.jsp

■ 게시글 수정 요청을 정상적으로 처리할 때 보여준다.

[board/WebContent/WEB-INF/view/modifySuccess.jsp]

```
01  <%@ page contentType="text/html; charset=utf-8"%>
02  <!DOCTYPE html>
03  <html>
04  <head>
05  <title>게시글 수정</title>
06  </head>
07  <body>
08
09  게시글을 수정했습니다.
10  <br>
11  ${ctxPath = pageContext.request.contextPath ; ''}
12  <a href="${ctxPath}/article/list.do">[게시글목록보기]</a>
13  <a href="${ctxPath}/article/read.do?no=${modReq.articleNumber}">[게시글내용보기]</a>
14  </body>
15  </html>
```

22.8 게시글 삭제 기능

■ 해당사항 없음