8장. Ajax, 게시판 댓글

Ajax

Ajax

- **Ajax란?**
  - Asynchronous JavaScript And XML의 약자이다.
  - "웹페이지 전체를 새로고침 하지 않고, 서버와 데이터를 주고받는 기술"이다
- **Ajax를 쓰는 이유**

  보통 HTML <form>을 쓰면 서버로 요청을 보낼 때 페이지가 새로 고침 된다.

  AJAX를 사용하면
  - 페이지 새로 고침 없이 요청 가능
  - 댓글이나 좋아요 같은 부분 업데이트에 유리
  - 사용자 경험(UX) 향상

- **Ajax 사용**

  예를 들어, 댓글을 등록하면

  $.ajax({...})

  이 코드가 실행되어 서버에 데이터를 보낸 뒤, 성공하면 페이지를 다시 로드하지 않고, 그 자리에서 바로 댓글 목록을 업데이트할 수 있다.

▪ RequestBody & ResponseBody

- 웹에서 화면전환(새로고침) 없이 이루어지는 동작들은 대부분 ***비동기 통신***으로 이루어진다.
- 비동기통신을 하기 위해서는 클라이언트에서 서버로 요청 메세지를 보낼 때, 본문에 데이터를 담아서 보내야 하고, 서버에서 클라이언트로 응답을 보낼 때에도 본문에 데이터를 담아서 보내야 한다. 이 본문이 바로 body 이다.

  즉, 요청 본문 **requestBody**, 응답 본문 **responseBody** 을 담아서 보내야 한다.

@RequestBody

이 어노테이션이 붙은 파라미터에는 http요청의 본문(body)이 그대로 전달된다.

json 기반의 메시지를 사용하는 요청의 경우에 이 방법이 매우 유용하다.

- 실습 예제

ajax 예제

01_get 요청(parameter 전달하기)

02_post 요청(parameter DTO로 받기)

03_post 요청(@RequestBody-json)

Ajax

- **index**

```
<title>ajax 메인..</title>
<style>
    a{
        display: block;
        margin: 10px;
    }
</style>
</head>
<body>
    <h2>ajax 실습 예제</h2>
    <div id="menu">
        <a th:href="@{/ajax/ex-01}">01_get 요청(parameter 전달하기)</a>
        <a th:href="@{/ajax/ex-02}">02_post 요청(parameter DTO로 받기)</a>
        <a th:href="@{/ajax/ex-03}">03_post 요청(@RequestBody-json)</a>
    </div>
</body>
```

Ajax

- **AjaxViewController**

```java
@RequestMapping("/ajax")
@Controller
public class AjaxViewController {

    @GetMapping("/ex-01")
    public String ajaxEx01() {
        return "ajax/ex-01";
    }

    @GetMapping("/ex-02")
    public String ajaxEx02() {
        return "ajax/ex-02";
    }

    @GetMapping("/ex-03")
    public String ajaxEx03() {
        return "ajax/ex-03";
    }
}
```

Ajax

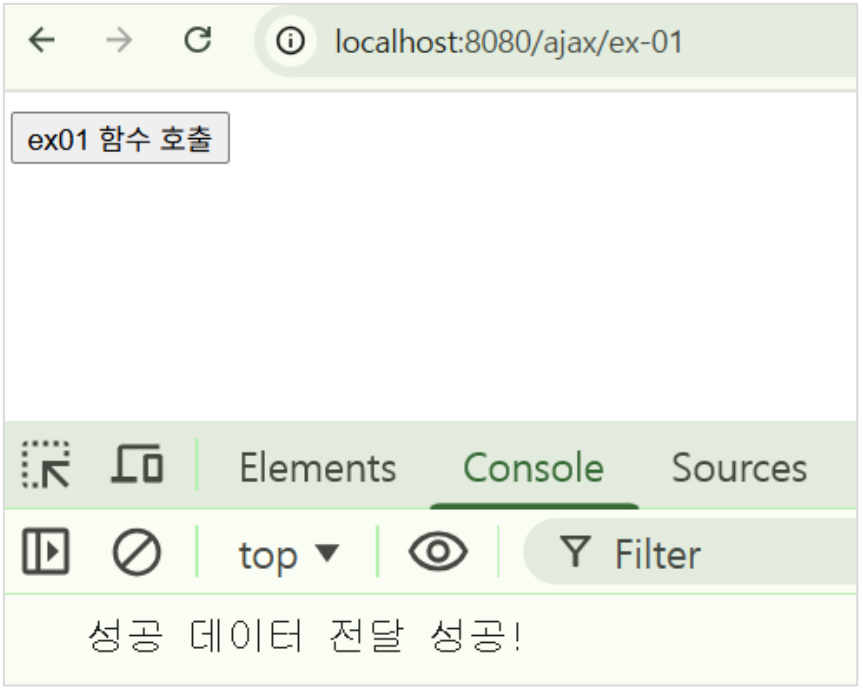
- **AjaxVController**

```java
@Slf4j
@Controller
public class AjaxController {

    @GetMapping("/ex01")
    public @ResponseBody String ex01(
                @RequestParam("param1") String param1,
                @RequestParam("param2") String param2) {
        log.info("param1 = " + param1 + ", param2 = " + param2);
        return "데이터 전달 성공!";
    }

    @PostMapping("/ex02")
    public @ResponseBody AjaxDTO ex02(@ModelAttribute AjaxDTO ajaxDTO) {
        log.info("ajaxDTO = " + ajaxDTO);
        return ajaxDTO;
    }
```

- **AjaxVController**

```
    //@RequestBody - json 데이터 요청
    @PostMapping("/ex03")
    public @ResponseBody AjaxDTO ex03(@RequestBody AjaxDTO ajaxDTO) {
        log.info("ajaxDTO = " + ajaxDTO);
        return ajaxDTO;
    }
}
```

- **ex01 구현**


localhost:8080/ajax/ex-01
ex01 함수 호출
Elements
Console
Sources
top
Filter
성공 데이터 전달 성공!

- **ex01.html**

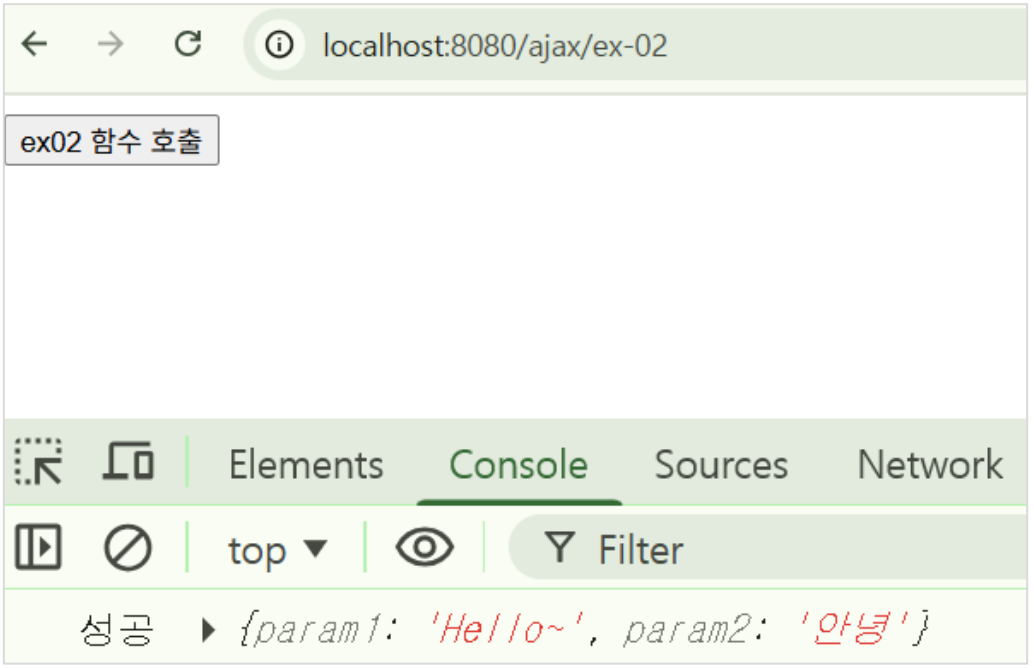
```
<title>ex01 페이지</title>
<script src="https://code.jquery.com/jquery-3.7.1.min.js"
    integrity="sha256-/JqT3SQfawRcv/BIHPThkBvs0OEvtFFmqPF/lYI/Cxo="
    crossorigin="anonymous"></script>
</head>
<body>
    <button onclick="ex01Fn()">ex01 함수 호출</button>
</body>

<script>
    const ex01Fn = () => {
        const text = "Hello~";
```

Ajax

- **ex01.html**

```
        $.ajax({
            //요청 방식: get, 요청 주소: /ex01, 요청 데이터: data
            type: "get",
            url: "/ex01",
            data: {
                "param1": text,
                "param2": "안녕"
            },
            //요청이 성공했을 때 실행되는 부분
            success: function(res){
                console.log("성공", res);
            },
            //요청이 실패했을 때 실행되는 부분
            error: function(){
                console.log("실패");
            }
        })
    }
</script>
```

- **ex02 구현**


localhost:8080/ajax/ex-02
ex02 함수 호출
Elements
Console
Sources
Network
top
Filter
성공
{param1: 'Hello~', param2: '안녕'}

- **ex02.html**

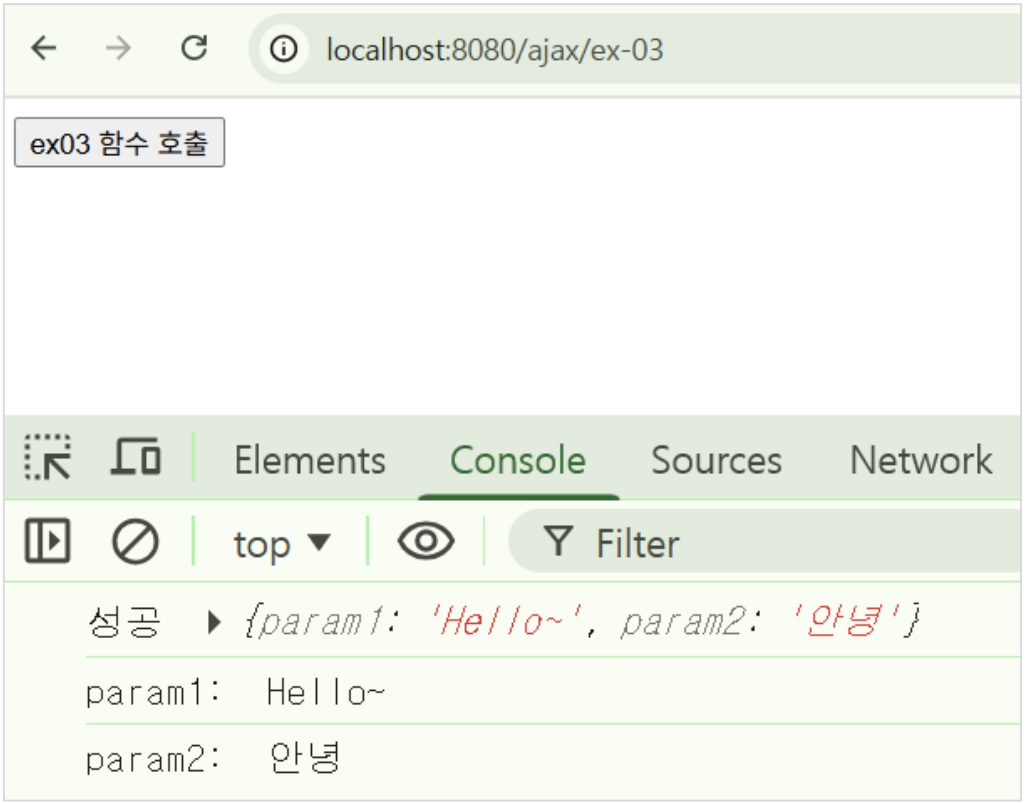
```
<body>
    <button onclick="ex02Fn()">ex02 함수 호출</button>
</body>

<script>
    const ex02Fn = () => {
        const text = "Hello~";

        const params = {
            "param1": text,
            "param2": "안녕"
        }
```

Ajax

- **ex02.html**

```
        $.ajax({
            //요청 방식: get, 요청 주소: /ex02, 요청 데이터: data
            type: "post",
            url: "/ex02",
            data: params,
            //요청이 성공했을 때 실행되는 부분
            success: function(res){
                console.log("성공", res);
            },
            //요청이 실패했을 때 실행되는 부분
            error: function(){
                console.log("실패");
            }
        })
    }
</script>
```

- **ex03 구현**


localhost:8080/ajax/ex-03
ex03 함수 호출
Elements
Console
Sources
Network
top
Filter
성공 {param1: 'Hello~', param2: '안녕'}
param1: Hello~
param2: 안녕

- **ex03.html**

```
<body>
    <button onclick="ex03Fn()">ex03 함수 호출</button>
</body>

<script type="text/javascript">
    const ex03Fn = () => {
        const text = "Hello~";

        const params = {
            "param1": text,
            "param2": "안녕"
        }
```

- **ex03.html**

```
        $.ajax({
            //요청 방식: get, 요청 주소: /ex03, 요청 데이터: data
            type: "post",
            url: "/ex03",
            data: JSON.stringify(params), //json 데이터로 변환
            contentType: "application/json", //컨텐트 타입 지정
            //요청이 성공했을 때 실행되는 부분
            success: function(res){
                console.log("성공", res);
                console.log("param1: ", res.param1);
                console.log("param2: ", res.param2);
            },
            //요청이 실패했을 때 실행되는 부분
            error: function(){
                console.log("실패");
            }
        })
    }
</script>
```

댓글 구현

글 상세 보기

가입 인사

안녕하세요
가입 인사 드려요~

글쓴이: 김기용(작성일: 2025-11-06 06:26:38.956278)

조회수: 18

댓글

댓글을 작성하려면 [로그인]하세요.

번호	작성자	내용	작성일
3	조하늘	행복하세요	2025-11-06 06:46
2	조하늘	반갑습니다. 좋은 활동 기대합니다.	2025-11-06 06:45
1	박단풍	어서오세요~ 축하합니다.	2025-11-06 06:28

댓글 구현

글쓴이: 김기용(작성일: 2025-11-06 06:26:38.956278)

조회수: 13

목록

댓글

조하늘

내용

등록

번호	작성자	내용	작성일
3	조하늘	행복하세요	2025-11-06 06:46
2	조하늘	반갑습니다. 좋은 활동 기대합니다.	2025-11-06 06:45
1	박단풍	어서오세요~ 축하합니다.	2025-11-06 06:28

댓글 구현

- **CommentDTO**

```
@Data
public class CommentDTO {
    private Long id;  //댓글 번호
    private String commentWriter;  //작성자
    private String commentContent; //내용
    private Long boardId; //게시글 번호
    private Timestamp commentDate; //작성일

    //entity를 dto로 변환
    public static CommentDTO toCommentDTO(Comment comment,
            Long boardId) {
        CommentDTO commentDTO = new CommentDTO();
        commentDTO.setId(comment.getId());
        commentDTO.setCommentWriter(comment.getCommentWriter());
        commentDTO.setCommentContent(comment.getCommentContent());
        commentDTO.setCommentDate(comment.getCommentDate());
        //commentDTO.setBoardId(comment.getBoard().getId());
        commentDTO.setBoardId(boardId);

        return commentDTO;
    }
}
```

댓글 구현

- **Comment - Entity**

```java
@Data
@Table(name = "t_comment")
@Entity
public class Comment {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(length=20, nullable=false)
    private String commentWriter;

    @Column
    private String commentContent;

    @CreationTimestamp
    private Timestamp commentDate;
```

댓글 구현

- **Comment - Entity**

```
    //Member 참조 관계(연관 매핑) : 다대일
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn
    private Member member;

    //Board 참조 관계(연관 매핑) : 다대일
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn
    private Board board;

    public static Comment toSaveEntity(CommentDTO commentDTO,
            Board board) {
        Comment comment = new Comment();
        comment.setCommentWriter(commentDTO.getCommentWriter());
        comment.setCommentContent(commentDTO.getCommentContent());
        comment.setBoard(board);
        return comment;

    }
}
```

댓글 구현

- **CommentRepository**

```
public interface CommentRepository extends JpaRepository<Comment, Long>{

    //select * from comment_table where board_id = ? order by id desc;
    List<Comment> findAllByBoardOrderByIdDesc(Board board);
}
```

댓글 구현

- **CommentController**

```
@RequestMapping("/comment")
@Controller
public class CommentController {

    private final CommentService commentService;

    //ResponseEntity - Body와 Header를 다룰수 있는 객체
    @PostMapping("/save")
    public ResponseEntity save(@ModelAttribute CommentDTO commentDTO) {
        Log.info("commentDTO: " + commentDTO);
        Long saveResult = commentService.save(commentDTO);
        if(saveResult != null) {
            //등록 성공하면 댓글 목록 가져와서 리턴
            List<CommentDTO> commentDTOList =
                    commentService.findAll(commentDTO.getBoardId());
            return new ResponseEntity<>(commentDTOList, HttpStatus.OK);
        }else {
            return new ResponseEntity<>("해당 게시글이 존재하지 않습니다.",
                    HttpStatus.NOT_FOUND);
        }

    }
}
```

댓글 구현

- **CommentService**

```
@RequiredArgsConstructor
@Service
public class CommentService {

    private final CommentRepository commentRepo;
    private final BoardRepository boardRepo;

    //댓글 등록
    public Long save(CommentDTO commentDTO) {
        //부모 엔티티 조회
        Optional<Board> optionalBoard =
                boardRepo.findById(commentDTO.getBoardId());
        if(optionalBoard.isPresent()) {
            Board board = optionalBoard.get();
            Comment comment = Comment.toSaveEntity(commentDTO, board);
            return commentRepo.save(comment).getId();
        }else {
            return null;
        }
    }
```

댓글 구현

- **CommentService**

```
    //댓글 목록
    public List<CommentDTO> findAll(Long boardId) {
        //해당 ID로 검색한 게시글 가져오기
        Board board = boardRepo.findById(boardId).get();
        List<Comment> commentList =
                commentRepo.findAllByBoardOrderByIdDesc(board);

        //entity를 dto로 변환
        List<CommentDTO> commentDTOList = new ArrayList<>();
        for(Comment comment : commentList) {
            CommentDTO commentDTO = CommentDTO.toCommentDTO(comment, boardId);
            commentDTOList.add(commentDTO);
        }

        return commentDTOList;
    }
}
```

댓글 구현

- **BoardController**

```
//글 상세보기
@GetMapping("/{id}")
public String getBoard(@PathVariable Long id,
        @PageableDefault(page=1) Pageable pageable,
        Model model) {
    try {
        //조회수 증가
        service.updateHits(id);
        //글 상세 보기
        Board board = service.findById(id);
        //댓글 목록 가져오기
        List<CommentDTO> commentDTOList = commentService.findAll(id);
        //모델 보내기
        model.addAttribute("board", board);
        model.addAttribute("page", pageable.getPageNumber());
        model.addAttribute("commentList", commentDTOList);
        return "board/detail";
    }catch(Exception e) {
        model.addAttribute("errorMsg", e.getMessage());
        return "error/errorPage";
    }
}
```

댓글 구현

- **글 상세보기 - detail.html**
  - Spring Security는 POST 요청 시 CSRF 보호(Cross-Site Request Forgery) 를 기본적으로 활성화합니다.
  - 즉, AJAX 요청에 CSRF 토큰이 없으면 **403 에러를 발생**시킵니다.

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org"
      xmlns:sec="http://www.thymeleaf.org/thymeleaf-extras-springsecurity6">
<head>
    <meta name="_csrf" th:content="${_csrf.token}"/>
    <meta name="_csrf_header" th:content="${_csrf.headerName}"/>
    <meta charset="UTF-8">
    <title>글 상세보기</title>
    <link rel="stylesheet" th:href="@{/css/style.css}">
    <script src="https://code.jquery.com/jquery-3.7.1.js" integrity="sha256-eKhay
</head>
<body>
    <section id="container">
        <h2>글 상세 보기</h2>
        <div class="boardDetail">
            <fieldset>
```

댓글 구현

- **detail.html**

```
<!-- 댓글 등록 -->
<div id="commentWrite">
    <h4>댓글</h4>
    <div sec:authorize="isAuthenticated()">
        <ul>
            <li>
                <!--  <input type="text" id="commentWriter" placeholder="작성자"> -->
                <div sec:authorize="isAuthenticated()">
                    <input type="text" id="commentWriter"
                        th:value="${#authentication.principal.name}">
                </div>
            </li>
            <li>
                <textarea id="commentContent" placeholder="내용"
                    rows="3" cols="50"></textarea>
            </li>
            <li>
                <button id="commentWriteBtn" onclick="commentWrite()">등록</button>
            </li>
        </ul>
    </div>
    <div sec:authorize="!isAuthenticated()">
        <p>댓글을 작성하려면 <a th:href="@{/members/login}">[로그인]</a>하세요.</p>
    </div>
</div>
```

댓글 구현

- **detail.html**

```
<!-- 댓글 목록 -->
<div id="commentList">
    <table class="commentList">
        <tr>
            <th>번호</th>
            <th>작성자</th>
            <th>내용</th>
            <th>작성일</th>
        </tr>
        <tr th:each="com: ${commentList}">
            <td th:text="${com.id}"></td>
            <td th:text="${com.commentWriter}"></td>
            <td th:text="${com.commentContent}"></td>
            <td th:text="${#dates.format(com.commentDate,
                    'yyyy-MM-dd HH:mm')}"></td>
        </tr>
    </table>
</div>
```

댓글 구현

- **detail.html**

```
<!-- 스크립트 영역 -->
<script th:inline="javascript">
    //CSRF 토큰 포함
    const token = $("meta[name='_csrf']").attr("content");
    const header = $("meta[name='_csrf_header']").attr("content");

    $(document).ajaxSend(function(e, xhr, options) {
        xhr.setRequestHeader(header, token);
    });

    //commentWrite() 정의
    const commentWrite = () => {
        const writer = document.getElementById("commentWriter").value;
        const content = document.getElementById("commentContent").value;
        const id = [[${board.id}]]
        //console.log("작성자: " + writer);
        //console.log("내용: " + content);
```

댓글 구현

- **detail.html**

```
$.ajax({
    //요청 방식: post, 요청 주소: /comment/save,
    //요청데이터: 작성자, 작성내용, 게시글 번호
    type: "post",
    url: "/comment/save",
    data: {
        "commentWriter": writer,
        "commentContent": content,
        "boardId": id
    },
    success: function(res){ //res = commentDTO
        console.log("요청 성공", res);
        console.log("요청 성공", res[0].commentWriter);
        console.log("요청 성공", res[0].commentContent);

        //댓글 목록 출력하기
        let output = "<table class='commentList'>";
        output += "<tr><th>번호</th>";
        output += "<th>작성자</th>";
        output += "<th>내용</th>";
        output += "<th>작성시간</th></tr>";
```

댓글 구현

- **detail.html**

```
                for(let i in res){
                    output += "<tr>";
                    output += "<td>" + res[i].id + "</td>";
                    output += "<td>" + res[i].commentWriter + "</td>";
                    output += "<td>" + res[i].commentContent + "</td>";
                    output += "<td>" + res[i].commentDate + "</td>";
                    output += "</tr>";
                }
                output += "</table>";
                document.getElementById('commentList').innerHTML = output;
                //댓글 내용 초기화
                document.getElementById('commentContent').value = "";
            },
            error: function(err){
                console.log("요청 실패", err);
            }
        });
    }
</script>
```