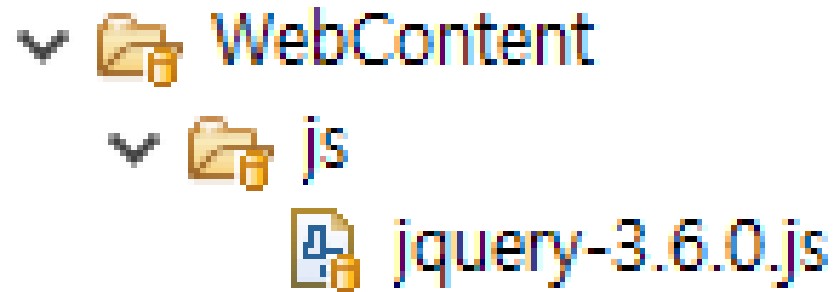


7주차 주말 미션

WEB - JQuery, Ajax 실습

4팀

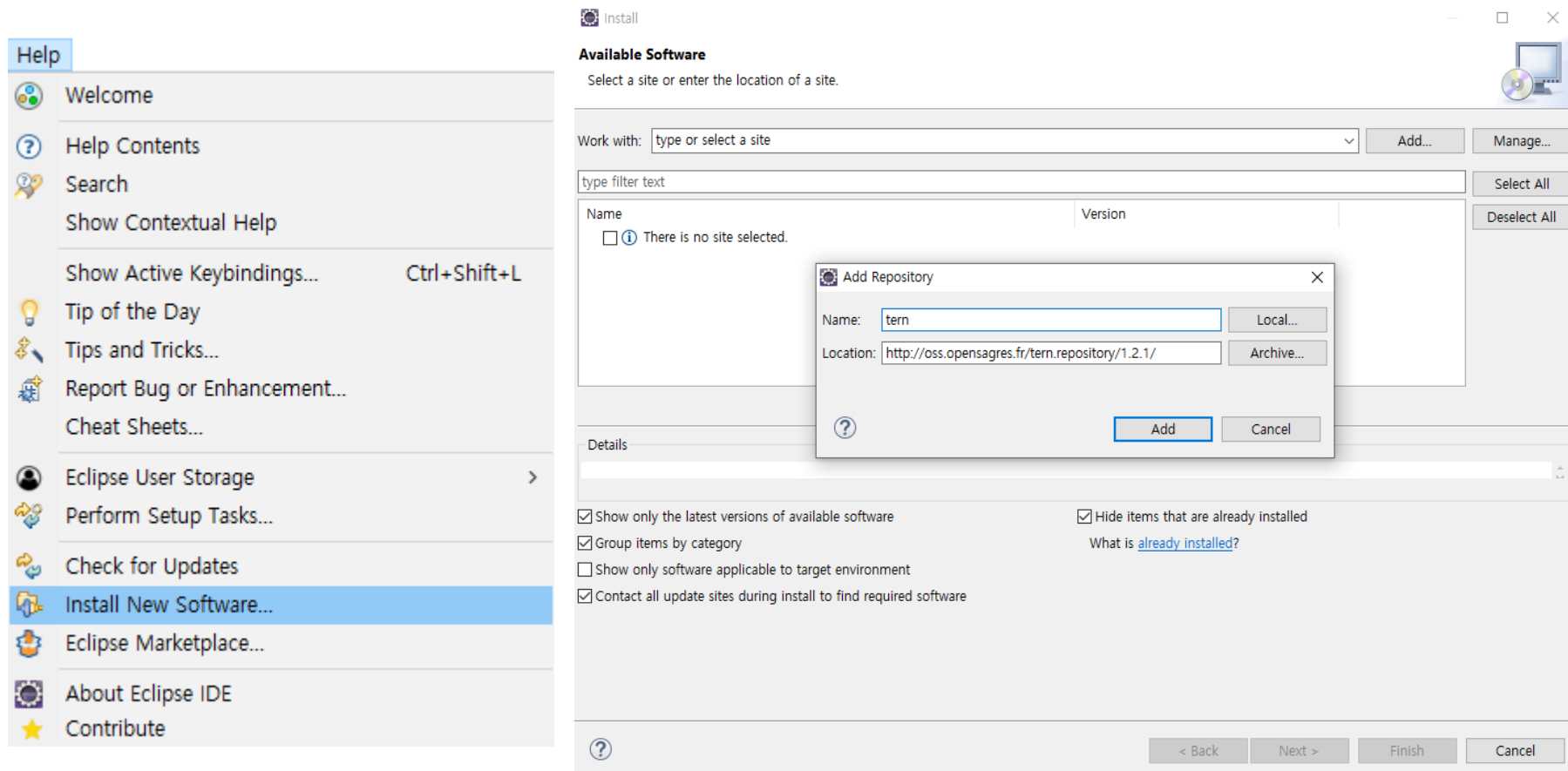
1. jquery-3.6.0.js 파일 등록(js라이브러리)



```
<meta charset="UTF-8">
<title>Insert title here</title>
<script type="text/javascript" src="js/jquery-3.6.0.js"></script>
</head>
```

제이쿼리를 사용하면 jsp, html에서 ajax를
사용해 서버로부터 비동기적으로 XML,
JSON형태의 데이터를 응답받는 과정을 쉽게
처리할 수 있음

2. tern 플러그인 등록(자동완성기능확장)



<http://oss.opensagres.fr/tern.repository/1.2.1/>

3. Json 라이브러리 등록

← → ↻ 🔒 mvnrepository.com/artifact/com.google.code.gson/gson/2.9.0

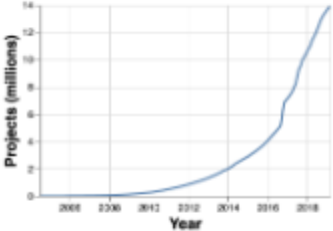
MVNREPOSITORY Search for groups, artifacts, categories Search

Home » com.google.code.gson » gson » 2.9.0

Gson » 2.9.0
Gson

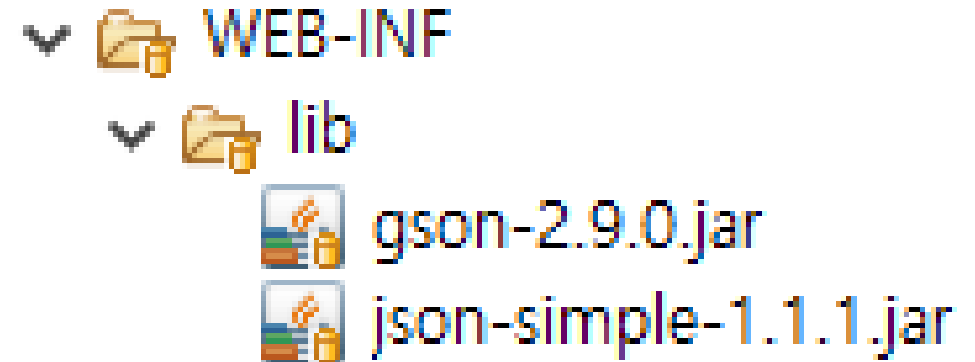
License	Apache 2.0
Categories	JSON Libraries
Date	(Feb 11, 2022)
Files	pom (7 KB) jar (243 KB) View All
Repositories	Central
Used By	17,712 artifacts

Indexed Artifacts (27.3M)



Popular Categories

- Aspect Oriented
- Actor Frameworks
- Application Metrics
- Build Tools



<https://mvnrepository.com/artifact/com.google.code.gson/gson>

JSON 데이터로 변환해 응답하기 위해 라이브러리를 사용함

JSON 사용하지 않을 경우

4-1. 아이디 중복체크 기능구현

[home](#) [방명록리스트](#) [회원가입](#)

회원가입

id : **사용불가**

name :

pw1 :

pw2 :

gender : ☒ 남자 ☐ 여자

☐ 독서 ☐ 요리 ☐ 음악감상

[home](#) [방명록리스트](#) [회원가입](#)

회원가입

id : **사용가능**

name :

pw1 :

pw2 :

gender : ☒ 남자 ☐ 여자

☐ 독서 ☐ 요리 ☐ 음악감상

```
<script type="text/javascript">
//아이디 유효성 검사
$("#user_id").blur(function() {
    var user_id = spaceCheck($("#user_id").val()); //공백제거
    $.ajax({
        url : "controller?cmd=idcheckAjaxAction",
        type: "post",
        data : {userId : user_id},
        success : function(resp) {
            if (resp == 1) {
                // 중복아이디
                $("#idCheck").text("사용불가");
                $("#idCheck").css("color", "red");
                $("#idCheck").css("font-size", "12pt");
                $("#idCheck").css("font-weight", "bold");
                $("#reg_submit").attr("disabled", true);
            }
            else if(resp==0){
                if (user_id.length ==0) {
                    $("#idCheck").text("아이디를 입력해주기 바람");
                    $("#idCheck").css("color", "red");
                    $("#idCheck").css("font-size", "12pt");
                    $("#idCheck").css("font-weight", "bold");
                    $("#reg_submit").attr("disabled", true);
                }
                else{
                    $("#idCheck").text("사용가능");
                    $("#idCheck").css("color", "green");
                    $("#idCheck").css("font-size", "12pt");
                    $("#idCheck").css("font-weight", "bold");
                    $("#reg_submit").attr("disabled", false);
                }
            }
        },
        error:function(){
            alert("통신실패");
        }
    });
});
</script>
```

5-1. 방명록 아이디 검색 기능 구현

[home](#) [방명록리스트](#) [회원가입](#)

[로그인 후 이용해주세요.]

방명록 리스트

작성글 검색 :

- 22 : admin - <공지사항> 유언비어 및 폭언 욕설 정치발언 금지 [2022-05-01]
- 3 : admin - 나 관리자다. [2022-04-28]

[home](#) [방명록리스트](#) [회원가입](#)

[로그인 후 이용해주세요.]

방명록 리스트

작성글 검색 :

- 10 : samsung - 갤럭시도 사야지 [2022-04-28]
- 4 : samsung - 갤럭시 사라 애들아 [2022-04-28]

```
<script type="text/javascript">
var searchList = function(){

    var user_id = $("#searchId").val();

    $.ajax({
        url : "controller?cmd=searchlistAjaxAction",
        type : "post",
        data : {userId : user_id},
        success : function(resp){
            $("#searchedList").html(resp);
        },error:function(){
            alert("통신실패");
        }
    });
};
</script>
```

6-1. SearchListAjaxAction.java

```
package ucamp.servlet;

import java.io.IOException;

public class SearchListAjaxAction implements Action {

    @Override
    public String action(HttpServletRequest request) throws ServletException, IOException {
        String url = "";
        String id = request.getParameter("userId");
        Collection<MemberVisitVO> list;
        if (id.length() != 0) {
            url = "searchedList.jsp";
            list = new MemberVisitDAO().searchList(id);
            if (list.size() != 0) {
                request.setAttribute("list", list);
            }
        } else {
            url = "searchedList.jsp";
            list = new MemberVisitDAO().getWrite();
            request.setAttribute("list", list);
        }
        return url;
    }
}
```


6-2. IdCheckAjaxAction.java

```
package ucamp.servlet;

import java.io.IOException;

public class IdCheckAjaxAction implements Action {

    @Override
    public String action(HttpServletRequest request) throws ServletException, IOException {
        String url = "signup.jsp";
        String id = request.getParameter("userId");
        if (id != null) {
            url = "idcheck.jsp";
            if (new MemberDAO().idCheck(id)) {
                System.out.println("이미 존재하는 아이디 " + id);
                request.setAttribute("data", 1);
            } else {
                System.out.println("사용가능 아이디 " + id);
                request.setAttribute("data", 0);
            }
        }
        return url;
    }
}
```

JSON 사용할 경우

4-2. 아이디 중복체크 기능구현

localhost:5432/WebTest/controller?cmd=addUserUI

home 로그인 회원가입

회원가입

id: choi 중복확인

name:

pw1:

pw2:

gender: ☐ 남 ☐ 여

☐ 독서 ☐ 요리 ☐ 영화감상

A

localhost:5432/WebTest/controller?cmd=addUserUI

home 로그인 회원가입 방명록보기

회원가입

id: 중복확인 사용불가

name:

pw1:

pw2:

gender: ☐ 남 ☐ 여

☐ 독서 ☐ 요리 ☐ 영화감상

A

DevTools Console

```
{inputId: 'choi', isValidId: 'false'}
```

```
<script type="text/javascript">
    $("#idDuplicateCheck").click(function(){
        var inputId = $('#inputId').val();
        $.ajax({
            url: "../ajaxController?cmd=checkDuplicateId",
            type: 'get',
            dataType: 'json',
            data: {"inputId": inputId},
            success: function(data){
                if(data.isValidId == "true"){
                    alert(inputId+"은(는) 사용할 수 있는 아이디입니다.");
                    $("#isValid").text("사용가능");
                    $("#isValid").css("color", "green");
                }else{
                    alert(inputId+"은(는) 이미 존재하는 아이디입니다.");
                    $('#inputId').focus();
                    $('#inputId').val('');
                    $("#isValid").text("사용불가");
                    $("#isValid").css("color", "red");
                }
                console.log(data);
            },
            error: function(request, status, error){
                console.log(request);
                alert("code:" + status);
                alert(error);
            }
        })
    })
</script>
```

5-2. 방명록 아이디 검색 기능 구현

localhost:5432/WebTest/controller?cmd=visitorListUI

home 로그인 회원가입 방명록보기

방명록 모두보기

작성자검색: 검색

번호 작성자아이디 작성일자

1	kim	4월 28, 2022
2	kim	4월 30, 2022
3	kim	4월 30, 2022
4	kim	4월 30, 2022
5	kim	4월 30, 2022

DevTools is now available in Korean!

Always match Chrome's language Switch DevTools to Korean Don't show again

Elements Console Sources Network Performance >> Filter Default levels No Issues 1 hidden

controller?cmd=visitorListUI:111

```
(5) [{"visitorSeq": 1, "memberId": "kim", "pw": "w123", "contents": "내용입니다", "writerDate": "2022-04-28"}, {"visitorSeq": 7, "memberId": "kim", "pw": "w123", "contents": "내용입니다222", "writerDate": "2022-04-30"}, {"visitorSeq": 8, "memberId": "kim", "pw": "w123", "contents": "내용입니다222", "writerDate": "2022-04-30"}, {"visitorSeq": 9, "memberId": "kim", "pw": "w123", "contents": "내용입니다222", "writerDate": "2022-04-30"}, {"visitorSeq": 15, "memberId": "kim", "pw": "w123", "contents": "내용입니다333", "writerDate": "2022-04-30"}]
```

localhost:5432/WebTest/controller?cmd=visitorListUI

home 로그인 회원가입 방명록보기

방명록 모두보기

작성자검색: 검색

번호 작성자아이디 작성일자

1	choi	4월 30, 2022
2	choi	4월 30, 2022
3	choi	4월 30, 2022
4	choi	4월 30, 2022

DevTools is now available in Korean!

Always match Chrome's language Switch DevTools to Korean Don't show again

Elements Console Sources Network Performance >> Filter Default levels No Issues 2 hidden

controller?cmd=visitorListUI:111

```
(5) [{"visitorSeq": 1, "memberId": "kim", "pw": "w123", "contents": "내용입니다", "writerDate": "2022-04-28"}, {"visitorSeq": 7, "memberId": "kim", "pw": "w123", "contents": "내용입니다222", "writerDate": "2022-04-30"}, {"visitorSeq": 8, "memberId": "kim", "pw": "w123", "contents": "내용입니다222", "writerDate": "2022-04-30"}, {"visitorSeq": 9, "memberId": "kim", "pw": "w123", "contents": "내용입니다222", "writerDate": "2022-04-30"}, {"visitorSeq": 15, "memberId": "kim", "pw": "w123", "contents": "내용입니다333", "writerDate": "2022-04-30"}]
```

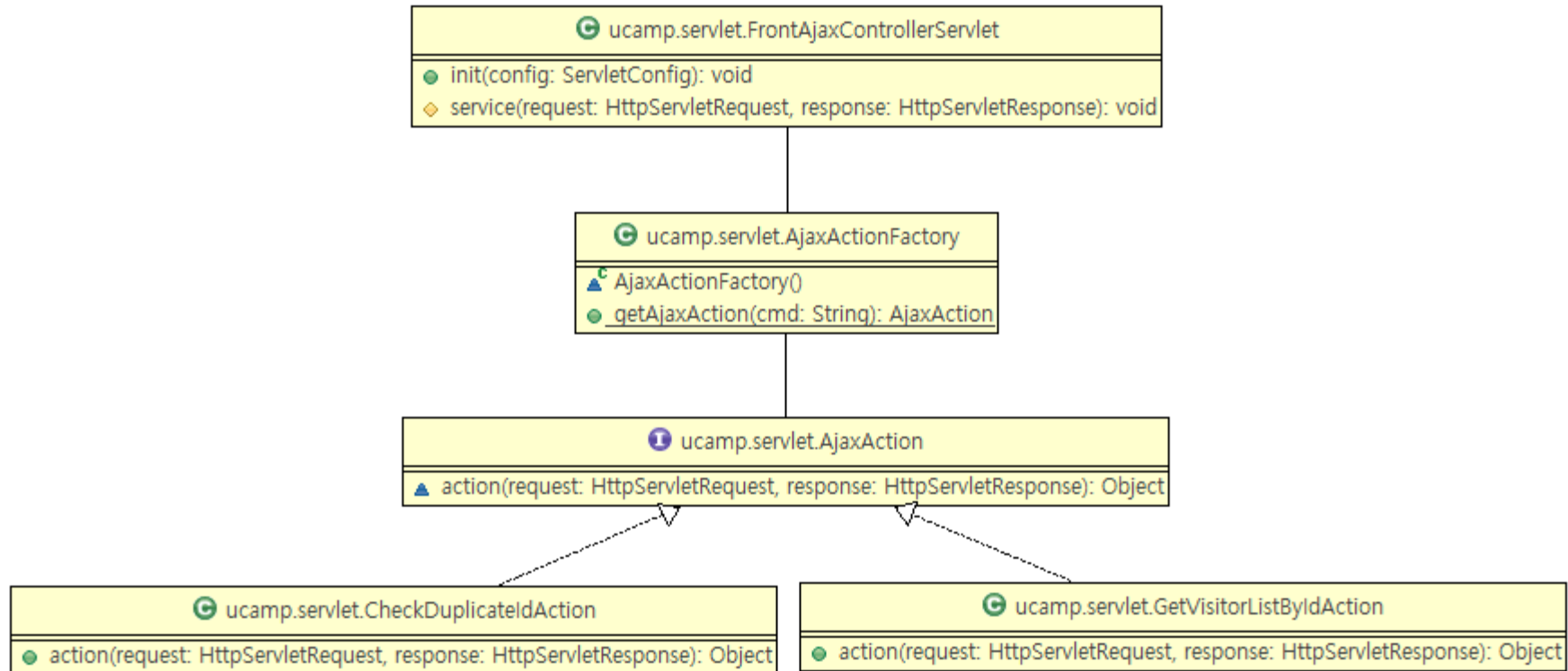
controller?cmd=visitorListUI:111

```
(4) [{"visitorSeq": 16, "memberId": "choi", "pw": "w123", "contents": "내용입니다", "writerDate": "2022-04-30"}, {"visitorSeq": 17, "memberId": "choi", "pw": "w123", "contents": "내용입니다222", "writerDate": "2022-04-30"}, {"visitorSeq": 18, "memberId": "choi", "pw": "w123", "contents": "내용입니다222", "writerDate": "2022-04-30"}, {"visitorSeq": 19, "memberId": "choi", "pw": "w123", "contents": "내용입니다333", "writerDate": "2022-04-30"}]
```

```
<script type="text/javascript">
$("#searchWriter").click(function(){
    alert('검색클릭');
    var inputId = $('#inputId').val();
    alert(inputId);

$.ajax({
    url: "./ajaxController?cmd=getVisitorListById",
    data: {"inputId": inputId},
    success: function(resp){
        console.log(resp);
        $("#vListTbody").html('');
        for(let i = 0; i<resp.length; i++){
            let tbody = '<tr onclick = \"location.href=\\\".\\" +
                +resp[i].visitorSeq+\\\".\">\">';
            tbody += '<td>' + (i+1) + '</td>'
            tbody += '<td>' + resp[i].memberId + '</td>'
            tbody += '<td>' + resp[i].writerDate + '</td>'
            tbody += '</tr>';
            //JS에서 html 태그 만드는 방법
            //.html() : 덮어쓰는형식
            //.after() : 뒤에추가해줌
            //.before() : 앞에추가해줌
            //.append() : 선택 태그 내부에 추가
            $('#vListTbody').append(tbody);
        }
    },
    error: function(request, status, error){
        console.log(request);
        alert("code:" + request.status);
        console.log(error);
    }
})
}</script>
```

6. Class Diagram



6-3. FrontAjaxController.java, AjaxAction.java, AjaxActionFactory.java

```
@WebServlet("/ajaxController")
public class FrontAjaxControllerServlet extends HttpServlet {

    @Override
    public void init(ServletConfig config) throws ServletException {
        super.init(config);
        ServletContext application = getServletContext();
        if(application.getAttribute("mDao") == null) {
            application.setAttribute("mDao", new MemberDAO());
        }
        if(application.getAttribute("vDao") == null) {
            application.setAttribute("vDao", new VisitorDAO());
        }
    }

    protected void service(HttpServletRequest request
        , HttpServletResponse response)
        throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        response.setContentType("text/json");
        response.setCharacterEncoding("utf-8");

        String cmd = request.getParameter("cmd");
        Object data = null;

        AjaxAction aa = null;
        if(cmd != null) {
            aa=AjaxActionFactory.getAjaxAction(cmd);
            data = aa.action(request, response);
        }

        response.getWriter().print(data);
    }
}
```

```
9 public interface AjaxAction {
10     /** 서블릿에서 request, response, data 제공*/
11     Object action(HttpServletRequest request
12         , HttpServletResponse response)
13         throws ServletException, IOException;
14 }
15

3 public class AjaxActionFactory {
4
5     AjaxActionFactory(){};
6     public static AjaxAction getAjaxAction(String cmd) {
7         AjaxAction aa = null;
8
9         switch(cmd) {
10             case "checkDuplicateId":
11                 aa = new CheckDuplicateIdAction();
12                 break;
13             case "getVisitorListById":
14                 aa = new GetVisitorListByIdAction();
15                 break;
16             default:
17                 break;
18         }
19
20         return aa;
21     }
22 }
23 }
```

6-4. CheckDuplicateIdAction.java

```
13 public class CheckDuplicateIdAction implements AjaxAction {
14
15     @Override
16     public Object action(HttpServletRequest request
17         , HttpServletResponse response)
18         throws ServletException, IOException {
19
20         String inputId = request.getParameter("inputId");
21
22         JsonObject data = new JsonObject();
23
24         data.addProperty("inputId", inputId);
25
26         MemberDAO mDao = (MemberDAO) request.getServletContext().getAttribute("mDao");
27         if(mDao.isValidAddMemberId(inputId)) {
28             data.addProperty("isValidId", "false");
29         }else {
30             data.addProperty("isValidId", "true");
31         }
32         return data;
33     }
34 }
```

6-5. GetVisitorListByIdAction.java

```
16 public class GetVisitorListByIdAction implements AjaxAction {
17
18     @Override
19     public Object action(HttpServletRequest request
20         , HttpServletResponse response)
21         throws ServletException, IOException {
22
23         String inputId = request.getParameter("inputId");
24
25         VisitorDAO vDao = (VisitorDAO) request.getServletContext().getAttribute("vDao");
26         Collection<VisitorVO> vList = vDao.getVisitorListById(inputId);
27
28         Gson gson = new Gson();
29         String json = gson.toJson(vList);
30
31         return json;
32     }
33
34 }
```


6-6. GetVisitorListByIdAction.java

```
16 public class GetVisitorListByIdAction implements AjaxAction {
17
18     @Override
19     public Object action(HttpServletRequest request
20         , HttpServletResponse response)
21         throws ServletException, IOException {
22
23         String inputId = request.getParameter("inputId");
24
25         VisitorDAO vDao = (VisitorDAO) request.getServletContext().getAttribute("vDao");
26         Collection<VisitorVO> vList = vDao.getVisitorListById(inputId);
27
28         Gson gson = new Gson();
29         String json = gson.toJson(vList);
30
31         return json;
32     }
33
34 }
```

7. 동기통신과 비동기통신 차이점 정리

• 동기 - '결과를 기다리는 것'

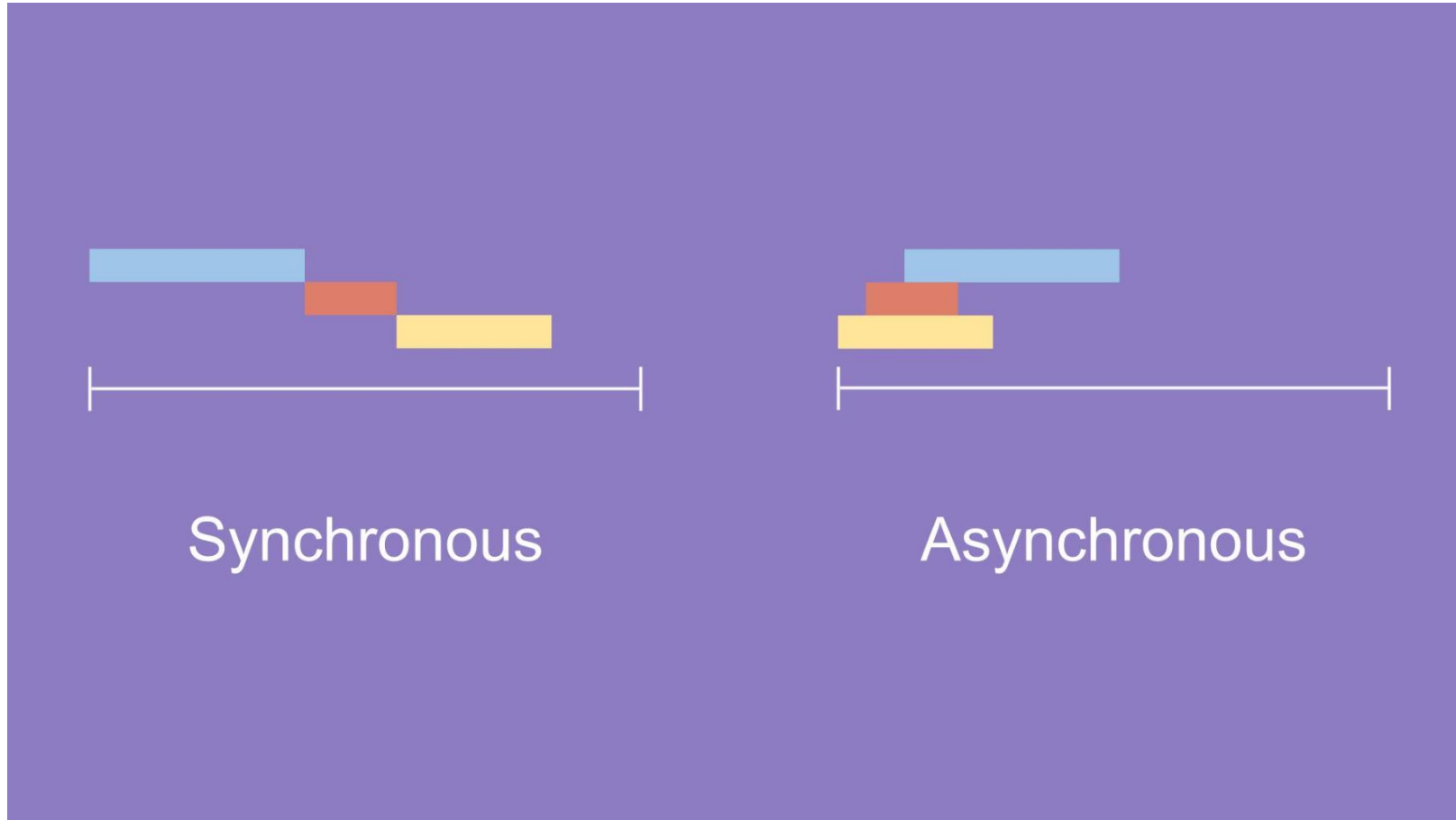
- 장점 : 업무가 단순
- - 단점 : 자원을 비효율적으로 사용
- --> 결과를 기다려야 하는 경우에 사용
- ex)은행 계좌이체, 회원가입, 온도체크

• 비동기 - '결과를 기다리지 않는 것'

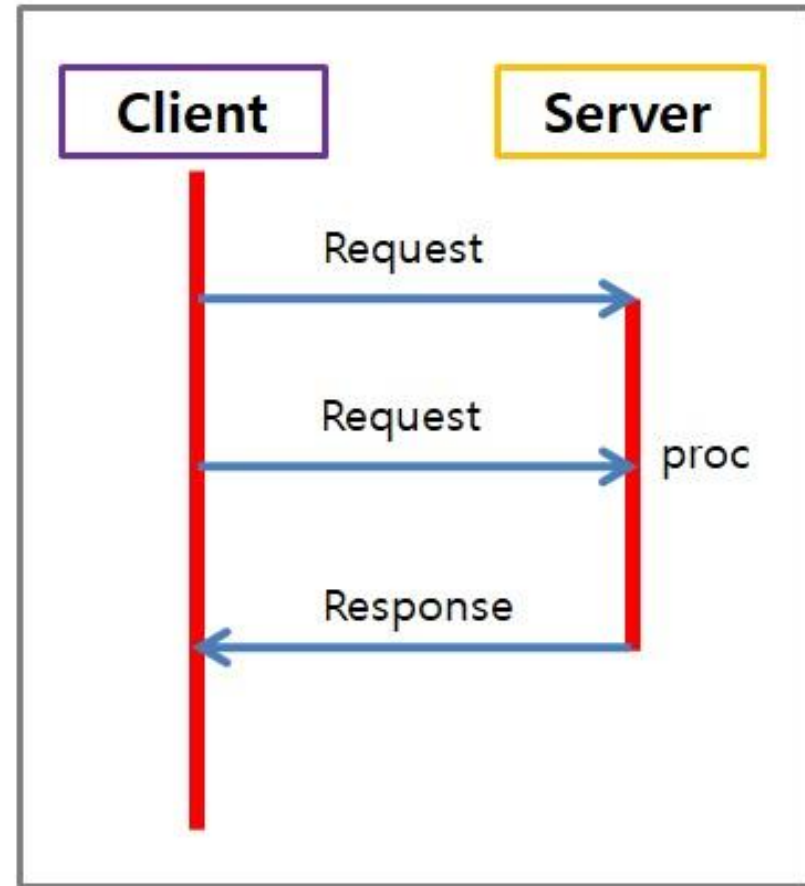
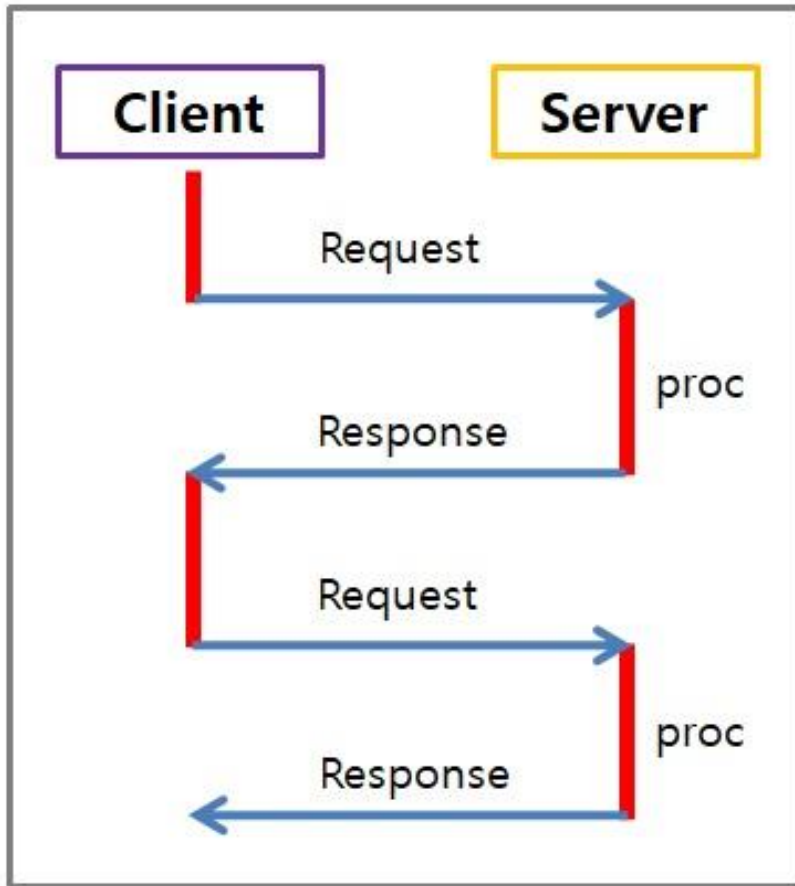
- 장점 : 자원을 효율적으로 사용
- 단점 : 업무가 복잡
- > 결과를 기다리지 않고 다른 작업과 병행할 때 사용하고 특정부분의 데이터만 변경할 때 사용
- ex) 아이디 중복체크, 라이브검색, 지도검색

- 동기방식은 설계가 매우 간단하고 직관적이지만 결과가 주어질 때까지 아무것도 못하고 대기해야 하는 단점이 있고, 비동기방식은 동기보다 복잡하지만 결과가 주어지는데 시간이 걸리더라도 그 시간 동안 다른 작업을 할 수 있으므로 자원을 효율적으로 사용할 수 있는 장점이 있다.
- 페이지 리로드의 경우, 전체 리소스를 다시 불러와야 하는데 이미지, 스크립트, 기타 코드 등을 모두 재요청할 경우 불필요한 리소스 낭비가 발생하게 되지만 비동기식 방식을 이용할 경우 필요한 부분만 불러와 사용할 수 있으므로 매우 큰 장점이 있다.
- 동기는 추구하는 같은 행위(목적)가 동시에 이루어지며, 비동기는 추구하는 행위(목적)가 다를 수도 있고, 동시에 이루어지지지도 않는다고 요약할 수 있겠다.

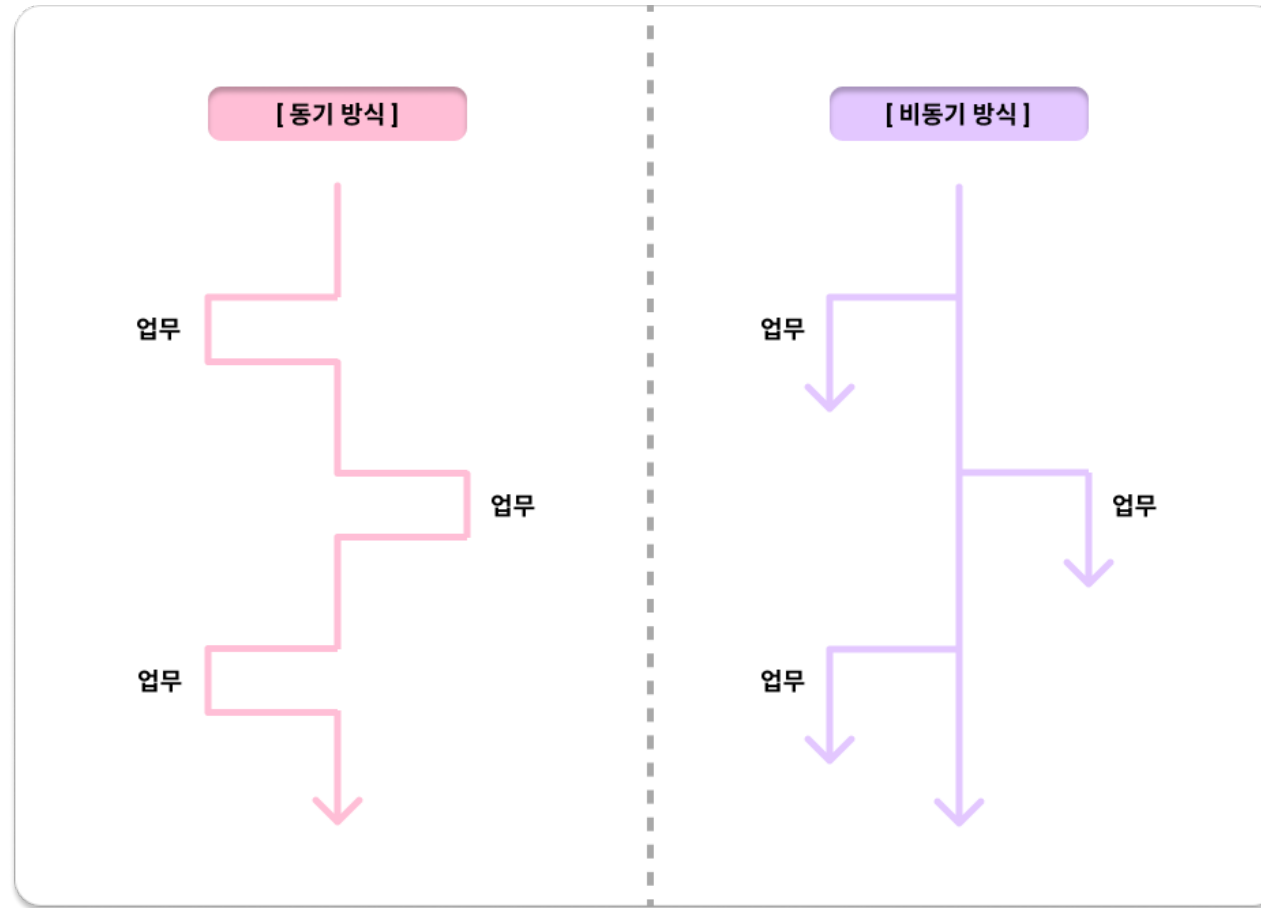
7-1. 동기화 비동기 참고사진1



7-2. 동기와 비동기 참고사진2



7-3. 동기와 비동기 참고사진3



8. 참고출처

- "22.03.17 java와 json파싱 Gson을 이용한 Ajax 통신", gogo_velog, 2022년 3월 17일 수정, 2022년 4월 30일 접속, <https://velog.io/@ggg4155/22.03.17-Ajax로-통신>
- "동기, 비동기 처리", goguard.log, 2020년 7월 8일 수정, 2022년 5월 1일 접속, <https://velog.io/@daybreak/동기-비동기-처리>.
- "동기와 비동기에 대한 간단요약", jch9537.log, 2020년 3월 15일 수정, 2022년 5월 1일 접속, <https://velog.io/@jch9537/비동기>.
- "AJAX란?", suyeon-b.github.io, 2021년 8월 12일 수정, 2022년 5월 1일 접속, <https://suyeon-b.github.io/javascript/2021/08/12/Ajax란.html>.

