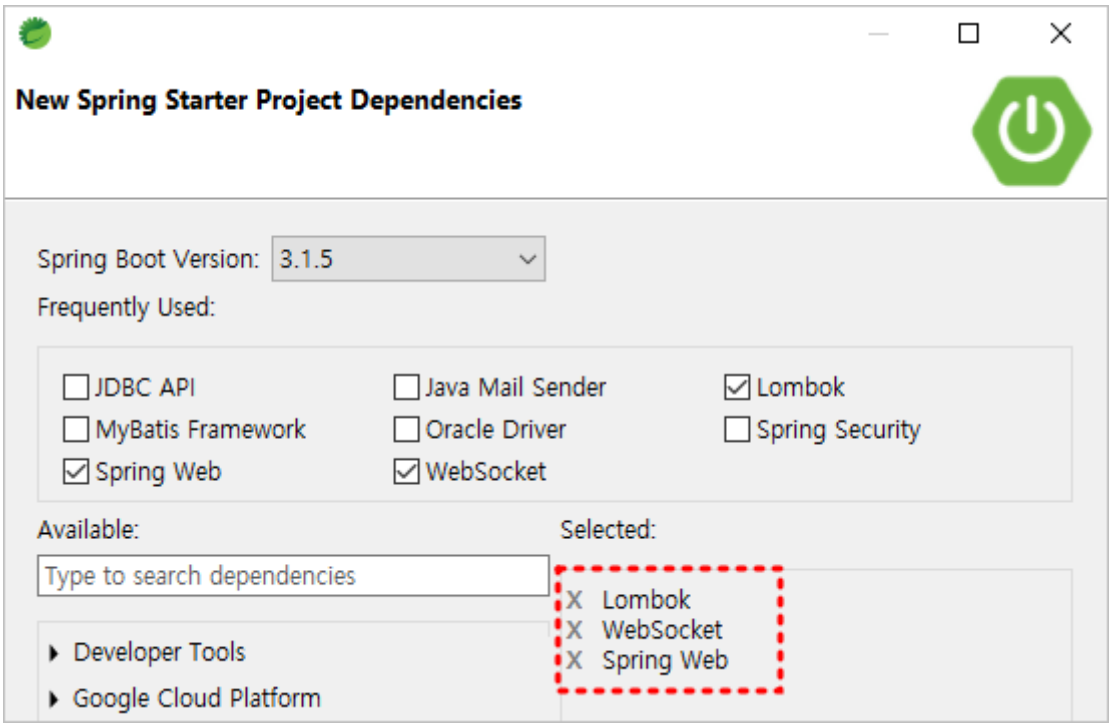
Websocket을 이용한 멀티채팅

프로젝트명 : B14WebSocket

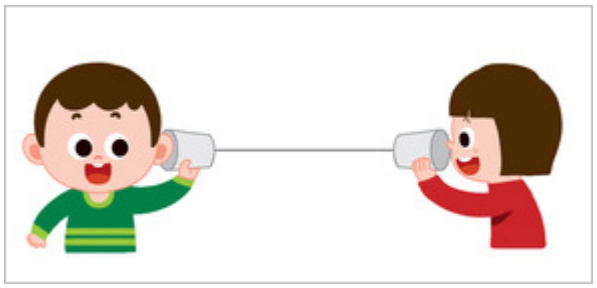
준비사항

- 의존설정 : Spring Web, Websocket, Lombok
- JSP 사용을 위한 설정을 한다.
- Refresh Gradle Project 를 눌러 적용한다.



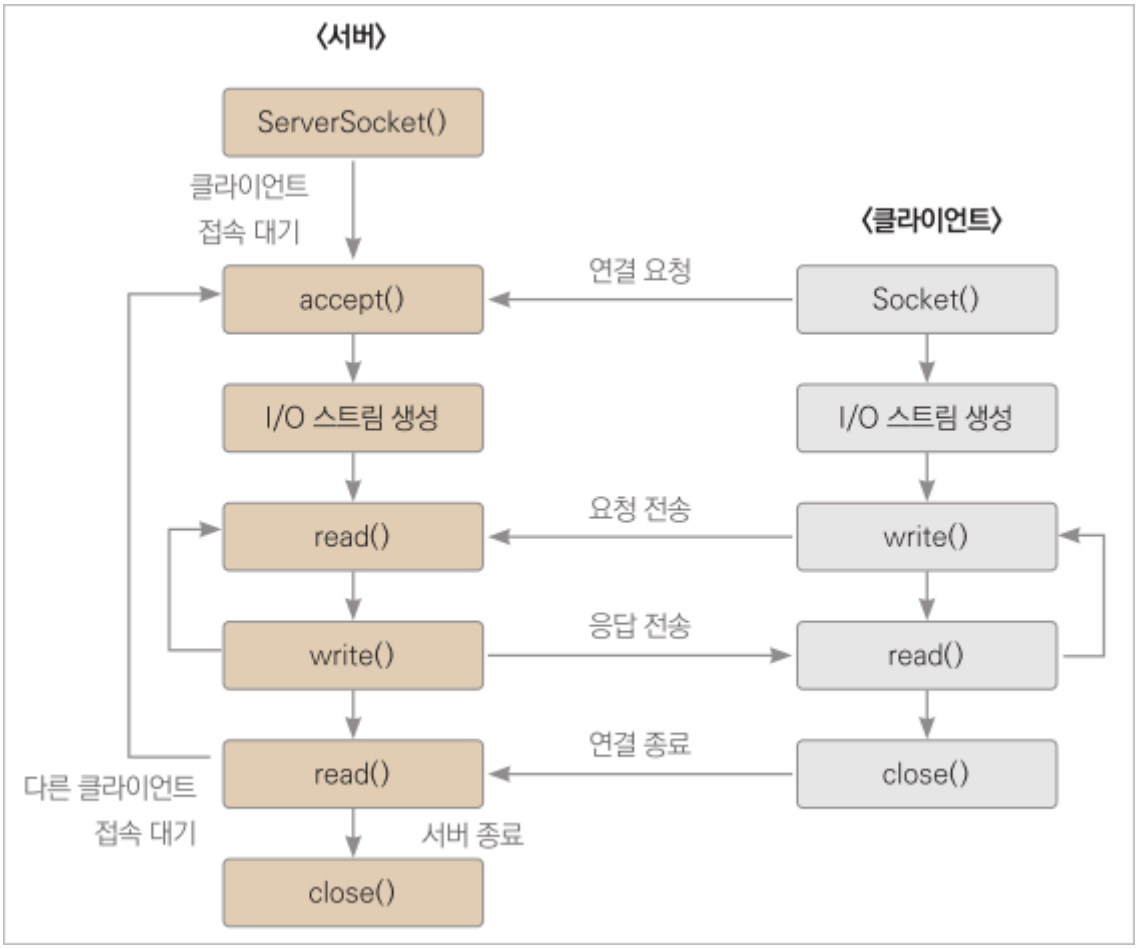
소켓이란..??

- 소켓(Socket)은 네트워크에서 동작하는 프로그램의 종착점(endpoint)이라고 주로 표현
- IP 주소와 포트 번호로 이루어져 있으며, 서버와 클라이언트가 양방향 통신을 할 수 있게 해주는 소프트웨어 장치



어린시절 가지고 놀던 종이컵 전화기에서 실은 네트워크의 역할을 하고, 종이컵이 바로 소켓의 역할을 담당한다.

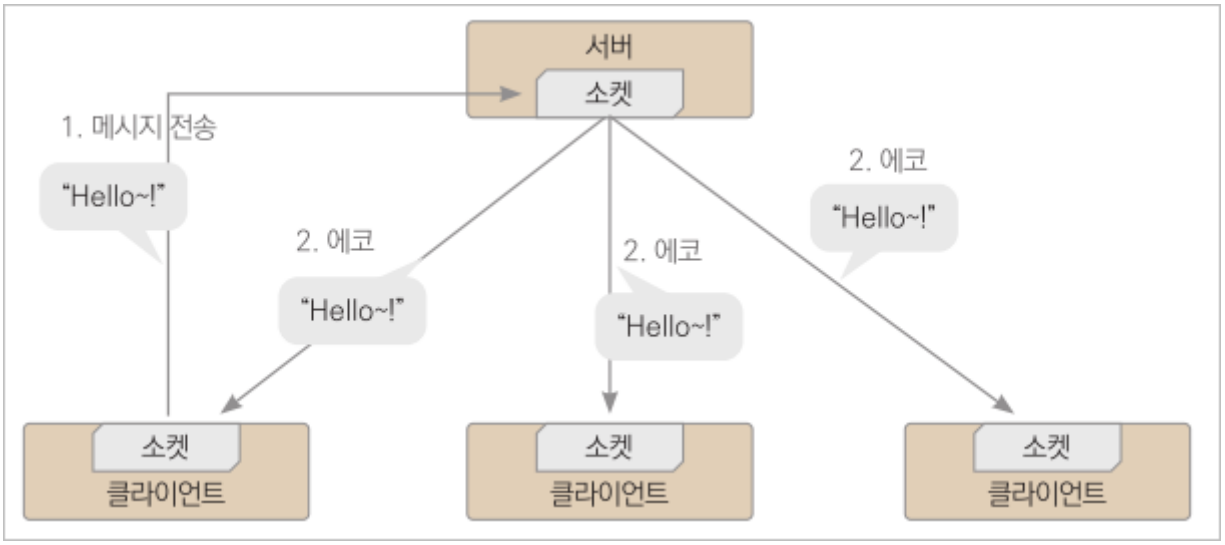
자바에서의 소켓 통신 절차



웹소켓이란..??

- 일반적인 웹 환경은 클라이언트의 요청을 받으면 응답 후 바로 연결을 종료하는 비연결(connectionless) 동기 소켓 방식을 사용
- 하지만 웹소켓(WebSocket)은 클라이언트의 요청에 응답한 후에도 연결을 그대로 유지하는 연결 지향(connection oriented) 방식을 사용
- 따라서 별도의 요청이 없어도 서버는 원하면 언제든 클라이언트로 데이터를 전송할 수 있다.

에코 클라이언트/서버 모델



뷰 : webapp/WEB-INF/views/home.jsp

```jsp
<body>
    <h2>웹소켓을 이용한 멀티채팅</h2>
    <ul>
        <li><a href="/">루트</a></li>
        <li><a href="/chatMain.do">멀티채팅</a></li>
    </ul>
</body>
</html>
```

컨트롤러 : com.edu.springboot.MainController.java

```java
    @RequestMapping("/")
    public String home() {
        return "home";
    }

    @RequestMapping("/chatMain.do")
    public String chatMain() {
        return "chatMain";
    }

    @RequestMapping("/chatWindow.do")
    public String chatWindow() {
        return "chatWindow";
    }
}
```

소켓설정 : com.edu.springboot.websocket.WebSocketConfig.java

```java
package websocket;

import org.springframework.context.annotation.Configuration;

@Configuration
@EnableWebSocket
@RequiredArgsConstructor
public class WebSocketConfig implements WebSocketConfigurer {

    private final WebSocketHandler webSocketHandler;

    @Override
    public void registerWebSocketHandlers(WebSocketHandlerRegistry registry) {
        registry.addHandler(webSocketHandler, "/myChatServer")
            .setAllowedOrigins("*");
    }
}
```

여기까지 작성하세요.

소켓핸들러 : com.edu.springboot.websocket.WebSocketHandler.java

```java
package websocket;

import java.io.IOException;

@Component
public class WebSocketHandler extends TextWebSocketHandler {

    private static final ConcurrentHashMap<String, WebSocketSession>
        CLIENTS = new ConcurrentHashMap<String, WebSocketSession>();

    @Override
    public void afterConnectionEstablished(WebSocketSession session)
            throws Exception {
        CLIENTS.put(session.getId(), session);
    }

    @Override
    public void afterConnectionClosed(WebSocketSession session,
            CloseStatus status)
                    throws Exception {
        CLIENTS.remove(session.getId());
    }
```

```java
    @Override
    protected void handleTextMessage(WebSocketSession session,
            TextMessage message)
                    throws Exception {
        String id = session.getId();
        CLIENTS.entrySet().forEach( arg->{
            if(!arg.getKey().equals(id)) {
                try {
                    arg.getValue().sendMessage(message);
                }
                catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

뷰 : /B14WebSocket/src/main/webapp/WEB-INF/views/chatMain.jsp

```jsp
<body>
    <script>
    function chatWinOpen() {
        var id = document.getElementById("chatId");
        if (id.value == "") {
            alert("대화명을 입력 후 채팅창을 열어주세요.");
            id.focus();
            return;
        }
        window.open("chatWindow.do?chatId=" + id.value, "",
                "width=320,height=400");
        id.value = "";
    }
    </script>
    <h2>웹소켓 채팅 - 대화명 적용해서 채팅창 띄워주기</h2>
    대화명 : <input type="text" id="chatId" />
    <button onclick="chatWinOpen();">채팅 참여</button>
</body>
```

뷰 : /B14WebSocket/src/main/webapp/WEB-INF/views/chatWindow.jsp (포트번호확인)

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<html>
<head>
<title>웹소켓 채팅</title>
<script>
var webSocket
    = new WebSocket("ws://localhost:8586/myChatServer");
var chatWindow, chatMessage, chatId;

window.onload = function() {
    chatWindow = document.getElementById("chatWindow");
    chatMessage = document.getElementById("chatMessage");
    chatId = document.getElementById('chatId').value;
}

function sendMessage() {
    chatWindow.innerHTML += "<div class='myMsg'>" +
                                chatMessage.value + "</div>"
    webSocket.send(chatId + '¦' + chatMessage.value);
    chatMessage.value = "";
    chatWindow.scrollTop = chatWindow.scrollHeight;
}
```

```
function disconnect() {
    webSocket.close();
}

function enterKey() {
    if (window.event.keyCode == 13) {
        sendMessage();
    }
}

webSocket.onopen = function(event) {
    chatWindow.innerHTML += "웹소켓 서버에 연결되었습니다.<br/>";
};

webSocket.onclose = function(event) {
    chatWindow.innerHTML += "웹소켓 서버가 종료되었습니다.<br/>";
};
```

```
webSocket.onerror = function(event) {
    alert(event.data);
    chatWindow.innerHTML += "채팅 중 에러가 발생하였습니다.<br/>";
};

webSocket.onmessage = function(event) {
    var message = event.data.split("¦");
    var sender = message[0];
    var content = message[1];
    if (content != "") {
        if (content.match("/")) {
            if (content.match(("/" + chatId))) {
                var temp = content.replace(("/" + chatId),
                        "[귓속말] : ");
                chatWindow.innerHTML += "<div>" + sender
                    + temp + "</div>";
            }
        }
```

```
        else {
            chatWindow.innerHTML += "<div>" + sender + " : "
                + content + "</div>";
        }
    }
    chatWindow.scrollTop = chatWindow.scrollHeight;
};
</script>
```

```
<style>
#chatWindow{border:1px solid black; width:270px; height:310px;
    overflow:scroll; padding:5px;}
#chatMessage{width:236px; height:30px;}
#sendBtn{height:30px; position:relative; top:2px; left:-2px;}
#closeBtn{margin-bottom:3px; position:relative; top:2px; left:-2px;}
#chatId{width:158px; height:24px; border:1px solid #AAAAAA;
    background-color:#EEEEEE;}
.myMsg{text-align:right;}
</style>
</head>
```

```
<body>
    대화명 :
    <input type="text" id="chatId" value="${ param.chatId }"
        readonly />
    <button id="closeBtn" onclick="disconnect();">채팅 종료</button>
    <div id="chatWindow"></div>
    <div>
        <input type="text" id="chatMessage" onkeyup="enterKey();">
        <button id="sendBtn" onclick="sendMessage();">전송</button>
    </div>
</body>
</html>
```