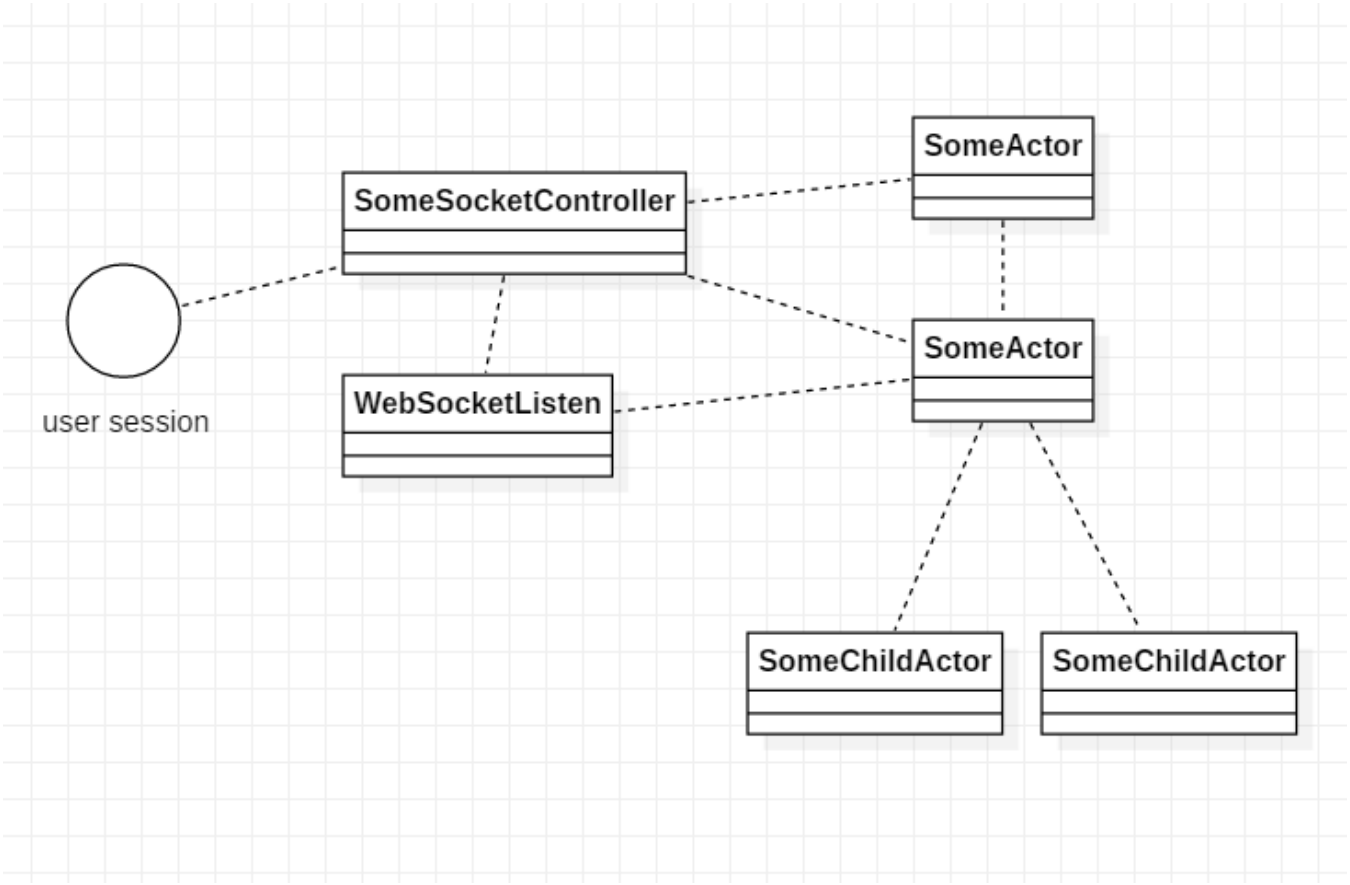
Spring with WebSocket



```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-websocket</artifactId>
</dependency>


<dependency>
   <groupId>org.webjars</groupId>
   <artifactId>webjars-locator-core</artifactId>
</dependency>
<dependency>
   <groupId>org.webjars</groupId>
   <artifactId>sockjs-client</artifactId>
   <version>1.0.2</version>
</dependency>

<dependency>
   <groupId>org.webjars</groupId>
   <artifactId>stomp-websocket</artifactId>
   <version>2.3.3</version>
</dependency>
```

Stomp Spring .

: https://spring.io/guides/gs/messaging-stomp-websocket/

```
public class WebSocketEventListener {
    @Autowired
    Lobby lobby;

    @Autowired
    private ActorSystem system;

    private static final Logger logger = LoggerFactory.getLogger(WebSocketEventListener.class);

    @Autowired
    private SimpMessageSendingOperations messagingTemplate;

    @EventListener
    public void handleWebSocketConnectListener(SessionConnectedEvent event) {
        logger.info("Received a new web socket connection");
        StompHeaderAccessor headerAccessor = StompHeaderAccessor.wrap(event.getMessage());
        String sessionId = headerAccessor.getUser().getName();
    }

    @EventListener
    void handleSessionConnectedEvent(SessionConnectedEvent event) {
        // Get Accessor
        StompHeaderAccessor sha = StompHeaderAccessor.wrap(event.getMessage());
        String sessionId = sha.getUser().getName();

        // OOP VS MESSAGE

        // #OOP - It's simple to develop locally, but many things need to change in order to be extended
remotely.
        lobby.addSender(sessionId,messagingTemplate);

        // #ACTOR - This can be extended to remote without implementation changes.
        ActorSelection lobby = system.actorSelection("/user/lobby");
        lobby.tell(new ConnectInfo(sessionId,messagingTemplate, ConnectInfo.Cmd.CONNECT), ActorRef.noSender());

        // Suppose you create several child actors under table.
        // You can send a message to all children with the following commands.
        // This is very useful.
        // Sameple Cmd : ActorSelection lobby = system.actorSelection("/user/table/*");
    }

    @EventListener
    public void handleWebSocketDisconnectListener(SessionDisconnectEvent event) {
        StompHeaderAccessor headerAccessor = StompHeaderAccessor.wrap(event.getMessage());
        String username = (String) headerAccessor.getSessionAttributes().get("username");
        String session = headerAccessor.getUser().getName();

        if(username != null) {
            logger.info("User Disconnected : " + username);
            GameMessage gameMessage = new GameMessage();
            gameMessage.setType(GameMessage.MessageType.LEAVE);
            gameMessage.setSender(username);
            //messagingTemplate.convertAndSend("/topic/public", gameMessage);

            // #OOP
            lobby.removeSender(session);

            // #ACTOR
            ActorSelection lobby = system.actorSelection("/user/lobby");
            lobby.tell(new ConnectInfo(session,messagingTemplate, ConnectInfo.Cmd.DISCONET), ActorRef.
noSender());
        }
    }

}
```

.

, .

:

- https://github.com/psmon/gameweb/blob/master/src/main/java/com/vgw/demo/gameweb/config/WebSocketConfig.java
- https://github.com/psmon/gameweb/blob/master/src/main/java/com/vgw/demo/gameweb/controler/ws/WebSocketEventListener.java

```
@Controller
@SuppressWarnings("Duplicates")
public class GameController {

    private static final Logger logger = LoggerFactory.getLogger(GameController.class);

    @MessageMapping("/game.req")
    @SendTo("/topic/public")
    public GameMessage gameReq(@Payload GameMessage gameMessage,
                               SimpMessageHeaderAccessor headerAccessor) {

        String  sessionId = headerAccessor.getUser().getName();
        logger.info("GameMsg:" + gameMessage );
        String gamePacket = gameMessage.getContent();
        String splitMessage[] = gamePacket.split("!!");

        String userName = headerAccessor.getSessionAttributes().get("username").toString();
        String userSession =  headerAccessor.getUser().getName();

        Object objTableNo = headerAccessor.getSessionAttributes().get("tableNo");
        Integer tableNo = objTableNo!=null? (Integer)objTableNo : -1;
....
}
```

.

, .

, .

:

- https://github.com/psmon/gameweb/tree/master/src/main/java/com/vgw/demo/gameweb/message
- https://github.com/psmon/gameweb/blob/master/src/main/java/com/vgw/demo/gameweb/controler/ws/GameController.java

JS

```
var stompClient = null;

function setConnected(connected) {
    $("#connect").prop("disabled", connected);
    $("#disconnect").prop("disabled", !connected);
    if (connected) {
        $("#conversation").show();
    }
    else {
        $("#conversation").hide();
        sceneControler('intro');
    }
    $("#greetings").html("");
}

function connect() {
    var socket = new SockJS('/ws');
```

```
    stompClient = Stomp.over(socket);
    stompClient.connect({}, function (frame) {
        setConnected(true);
        console.log('Connected: ' + frame);

        // for broad cast
        stompClient.subscribe('/topic/public', onMessageReceived );

        // for send to some
        stompClient.subscribe('/user/topic/public', onMessageReceived );

        var username=$("#name").val();
        if(username.length<1){username="Unknown"};
        // Tell your username to the server
        stompClient.send("/app/lobby.addUser",
            {},
            JSON.stringify({sender: username, type: 'JOIN'})
        )
    });
}

function disconnect() {
    if (stompClient !== null) {
        stompClient.disconnect();
    }
    setConnected(false);
    console.log("Disconnected");
}

function joinTable(tableNo) {
    stompClient.send("/app/game.req",
        {},
        JSON.stringify({content: 'join',num1:tableNo, type: 'GAME'})
    )
}

function seatTable() {
    stompClient.send("/app/game.req",
        {},
        JSON.stringify({content: 'seat', type: 'GAME'})
    )
}


function sendChatMsg() {
    var content = $('#gamemsg').val();
    stompClient.send("/app/hello",
        {},
        JSON.stringify({content: content, type: 'CHAT'})
    )
}

function sendGameMsg() {
    var content = $('#gamemsg').val();
    stompClient.send("/app/game.req",
        {},
        JSON.stringify({content: content, type: 'GAME'})
    )
}

function sendGameAction(action) {
    var content = $('#gamemsg').val();
    stompClient.send("/app/game.req",
        {},
        JSON.stringify({content: action.content,num1:action.num1,num2:action.num2, type: 'ACTION'})
    )
}

function showGreeting(message) {
    $("#greetings").append("<tr><td>" + message + "</td></tr>");
}
```

```
function onMessageReceived(payload) {
    var message = JSON.parse(payload.body);

    var messageElement = document.createElement('li');

    if(message.type == 'JOIN') {
        showGreeting('Welcome ' + message.sender)
    } else if (message.type == 'LEAVE') {
        showGreeting(message.sender + 'left!')
    } else if(message.type == 'GAME'){
        messageControler(message);
        //showGreeting(message.content);
    } else{
        showGreeting(message.content);
    }
}

$(function () {
    $("form").on('submit', function (e) {
        e.preventDefault();
    });
    $( "#connect" ).click(function() { connect(); });
    $( "#disconnect" ).click(function() { disconnect(); });
    $( "#send" ).click(function() { sendGameMsg(); });


});
```

.js .

4 .

- connect :
- disconnect :
- onMessage :
- send :

:

- https://github.com/psmon/gameweb/blob/master/src/main/resources/static/app.js