


WebSocket tutorial for Java

Dr. Balázs Simon (sbalazs@iit.bme.hu), BME IIT, 2023

1 Introduction

This document describes how to create a Hello World WebSocket service using Eclipse and WildFly.

2 Application

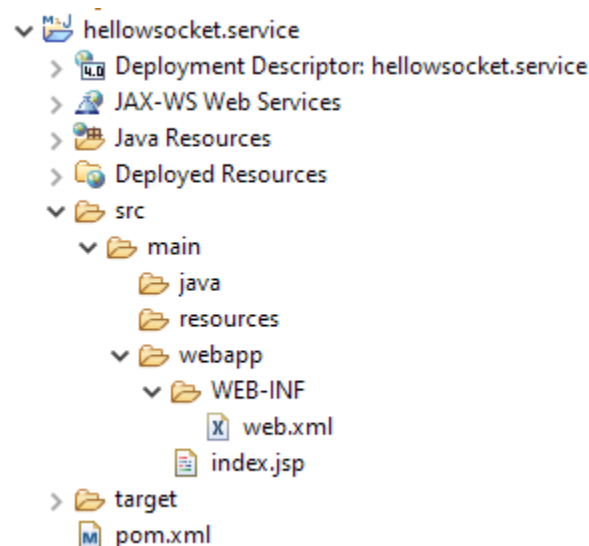
Open a command prompt from the Eclipse workspace directory and issue the following **mvn** command to create a new web application (the  symbol means that the command is continued in the next line, and the whole command should be typed as a single line):

```
c:\Users\[user]\eclipse-soi>mvn archetype:generate -DgroupId=soi   
-DartifactId=helloworld.service   
-DarchetypeArtifactId=maven-archetype-webapp -DinteractiveMode=false
```

(Make sure to use the simple command prompt and not PowerShell, since PowerShell handles the dash switches differently.)

Replace the **pom.xml** in the newly created **helloworld.service** folder with the one attached to this tutorial (**helloworld.service/pom.xml**).

Import the project into Eclipse and set the **Dynamic Web Module** version to **5.0** on the **Project Facets** property page. Also, create a folder called **java** under **src/main**. (See the Web Service tutorial for more details). The project should look like this:



Add a new class called **HelloEndpoint** in the **src/main/java** folder under the package **hellowsocket.service** with the following content:

```
package hellowsocket.service;

import jakarta.websocket.OnClose;
import jakarta.websocket.OnError;
import jakarta.websocket.OnMessage;
import jakarta.websocket.OnOpen;
import jakarta.websocket.Session;
import jakarta.websocket.server.ServerEndpoint;

@ServerEndpoint("/hello")
public class HelloEndpoint {
    @OnOpen
    public void open(Session session) {
        System.out.println("WebSocket opened: " + session.getId());
    }

    @OnClose
    public void close(Session session) {
        System.out.println("WebSocket closed: " + session.getId());
    }

    @OnError
    public void error(Throwable t) {
        System.out.println("WebSocket error: " + t.getMessage());
    }

    @OnMessage
    public String message(String msg) {
        System.out.println("WebSocket message: " + msg);
        return "Hello: "+msg;
    }
}
```

Add a new HTML file called **hello.html** under the **webapp** folder with the following content:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Hello World WebSocket client</title>
    <script>
        var wsUrl = getRootUri() + "/hellowsocket.service/hello";
        var websocket = null;

        function getRootUri() {
            return "ws://" +
            (document.location.hostname == "" ? "localhost" : document.location.hostname)
            + ":" +
            (document.location.port == "" ? "8080" : document.location.port);
        }
    </script>
</head>
</html>
```

```

function init() {
    output = document.getElementById("output");
}

function initWebSocket() {
    websocket = new WebSocket(wsUrl);
    websocket.onopen = function(evt) {
        onOpen(evt);
        doSend();
    };
    websocket.onmessage = function(evt) {
        onMessage(evt);
    };
    websocket.onerror = function(evt) {
        onError(evt);
        websocket = null;
    };
    websocket.onclose = function(evt) {
        onClose(evt);
        websocket = null;
    };
}

function send_message() {
    if (websocket == null) {
        initWebSocket();
    } else {
        doSend();
    }
}

function onOpen(evt) {
    writeToScreen("Connected to endpoint.");
}

function onMessage(evt) {
    writeToScreen("Message received: " + evt.data);
}

function onError(evt) {
    writeToScreen('<span style="color: red;">ERROR:</span> ' + evt.data);
}

function onClose(evt) {
    writeToScreen("Connection closed.");
}

function doSend() {
    message = textID.value;
    websocket.send(message);
    writeToScreen("Message Sent: " + message);
}

```

```

    function writeToScreen(message) {
        var pre = document.createElement("p");
        pre.style.wordWrap = "break-word";
        pre.innerHTML = message;

        output.appendChild(pre);
    }

    window.addEventListener("load", init, false);

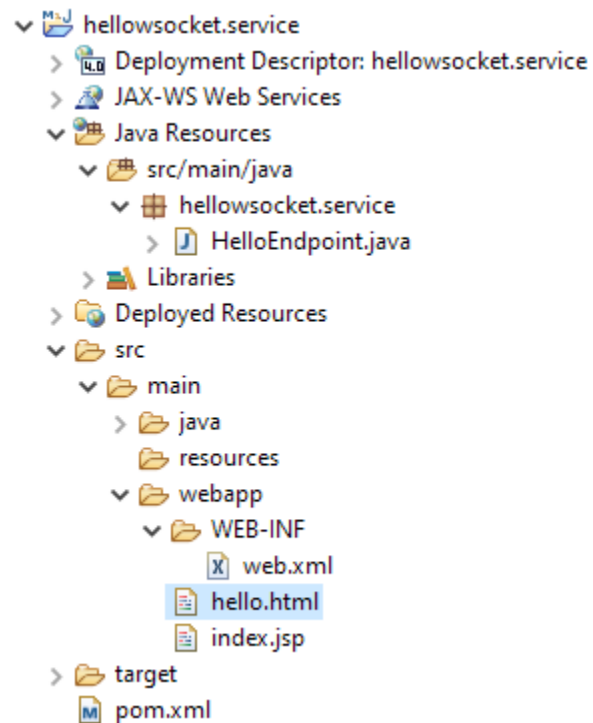
</script>
</head>
<body>
    <h1 style="text-align: center;">Hello World WebSocket client</h1>

    <br/>

    <div style="text-align: center;">
        <form action="">
            <input onclick="send_message()" value="Send" type="button"/>
            <input id="textID" name="message" value="me" type="text"/><br/>
        </form>
    </div>
    <div id="output"></div>
</body>
</html>

```

The project should look like this:

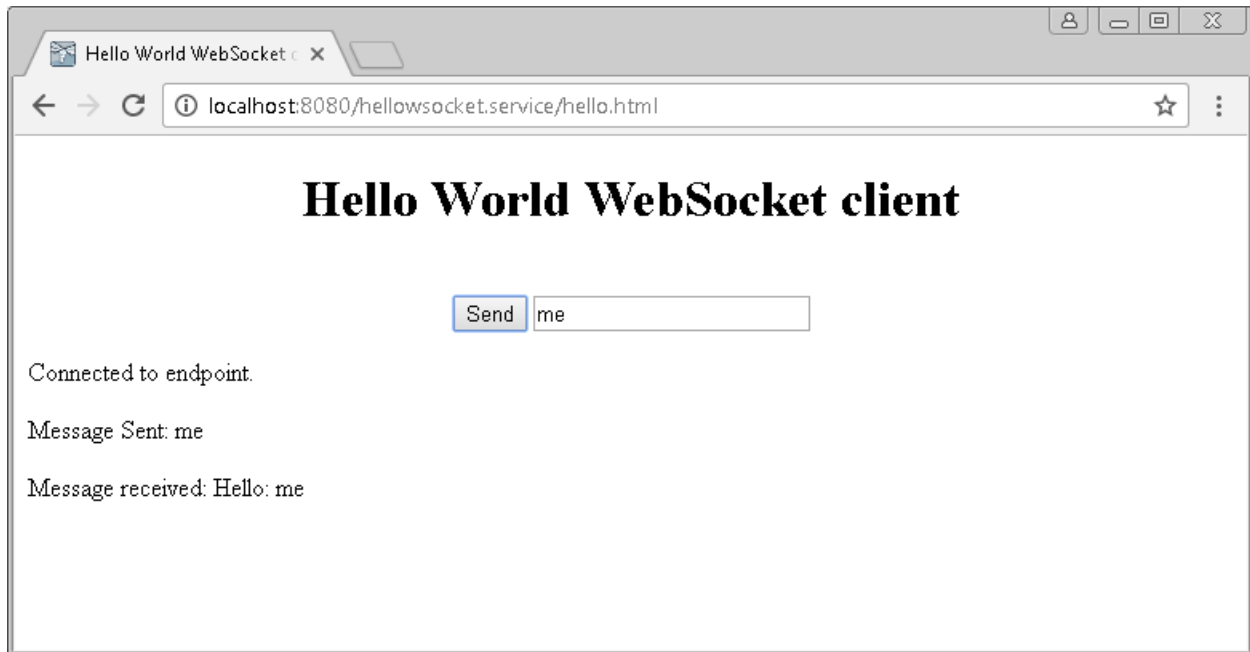


Right click on the project and deploy it to the WildFly server. It should deploy without errors.

Open the following URL in a browser to test the application:

`http://localhost:8080/hellowsocket.service/hello.html`

Type something in the input field and click on the **Send** button:



Also, in the server log there should be:

