Lesson 18
Prototyping Online Components for using the REST and WebSockets

Drawing an Implementation table for web APIs using REST Methods

- Using REST methods in client or server, first making a table makes the coding easy

Drawing an Implementation table for web APIs using REST Methods

- Column 1 of each row has the URL for communication for implementing an action
- Column 2 may specify the action (REST methods) such as PUT/ GET/ POST/ DELETE for implementation.

Drawing an Implementation table for web APIs using REST Methods

- Column 3 may specify for each action, the authentication code or method (device platform ID, such as MAC address) for secured communication to and from other end (server or Application)
- Password can be a code internally generated at device platform using some algorithm using a secret key input (Example 9.8)

Drawing an Implementation table for web APIs using REST Methods

- Column 4 specifies the API inputs/parameters for initiating the action.
- Column 5 specifies the API outputs for the inputs. The outputs communicate to other end and initiate the execution of methods, callback functions, generate Cookie or response

Web APIs using WebSocket Methods

- HTTP client first pulls data from one end to another and later other end-clients in other direction
- WebSocket enables bi-directional communication at the same instances between two ends.
- The webSocket needs lesser header-size when compared to HTTP header size (Figure 3.9)

Eclipse Jetty WebSocket (String)

- A WebSocket implementation in Java and is open source from a web link https://www.eclipse.org/websocket
- Declare maximum message size declares as WebSocket (maxTextMessageSize)
- WebSocket Client and server both use the callback listeners onOpen, onMessage

Eclipse Jetty import for Java implementations of org.eclipse.jetty. Websocket Client and APIs

- websocket.client.WebSocketClient;
- websocket.api.Session;
- websocket.api.StatusCode;
- websocket.api.annotations.OnWebSocketOpen;

Eclipse Jetty import for Java implementations of org.eclipse.jetty.websocket.api.annotations

- OnWebSocketConnect;
- OnWebSocketMessage;
- OnWebSocketClose;
- import.OnWebSocketOnError;

The callback WebSocket methods

- onWebSocketConnect()
- onWebSocketMessage()
- onWebSocketClose()

Open source Paho Go Client at Eclipse Paho

- Consists of added WebSocket (ws) clients to connect to MQTT broker
- Open source Eclipse Ponte consists of added MQTT-over-ws clients in JavaScript

Open source client-server WebSocket methods

- For implementation on Arduino device platform can be used from the web links https://github.com/krohling/ArduinoWebsocket

and https://github.com/djsb/arduino-websocket. Krohling Arduino ws client and ws server (bi-directional) implementation

WebSocket clients for bi-directional exchange of messages

- wsclient.available(), wsclient.connect(Server)
- wsClient.connection(), wsClient.connected(),
- wsClient.send(), wsClient.setDataArrived() and
- wsClient.setDataArrivedDelegate(String dataArrived) for communication after creating WebSocket clients

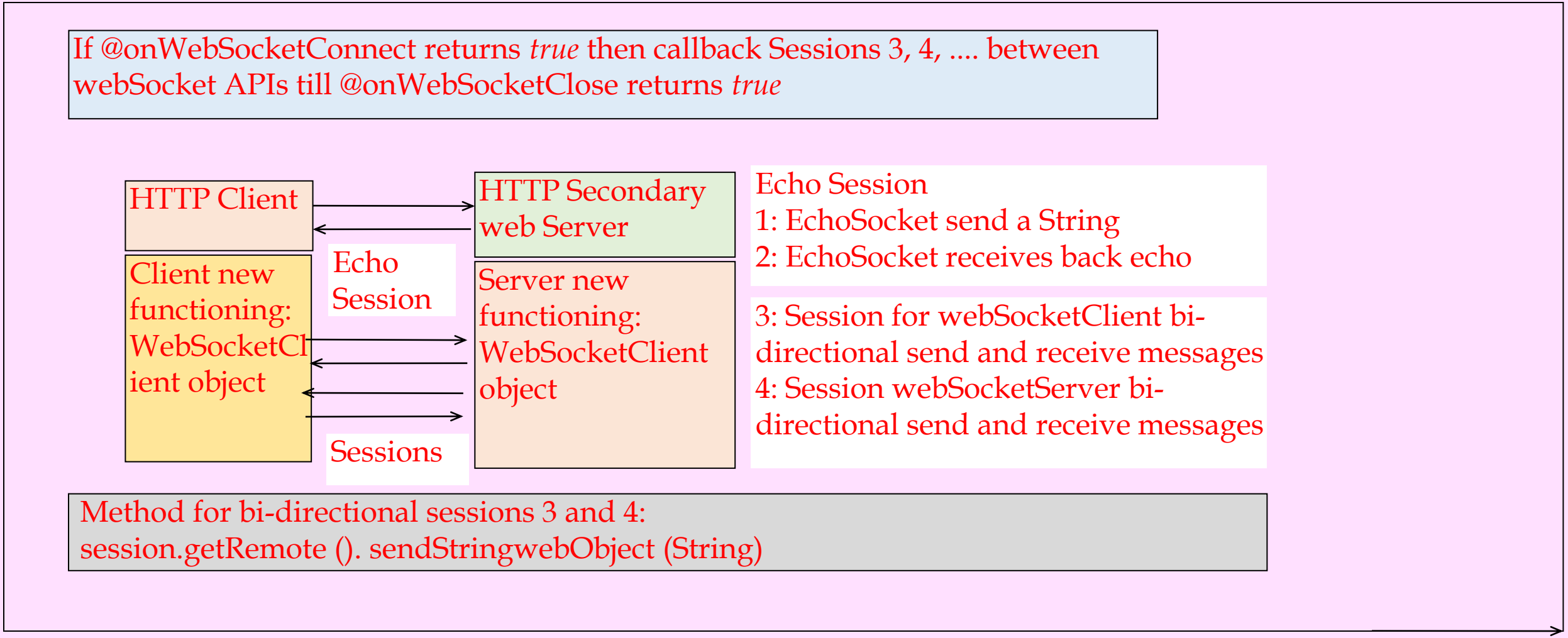

Fig. 9.6 Sequences of Sessions for message exchanges when using the webSockets

Drawing a Table to code for the sequences of sessions using WebSockets

- Column 1 of each row has the URI or URL of message destination end for communication for implementing an action

Drawing a Table to code for the sequences of sessions using WebSockets

- Column 2 may specify the action such as following callback WebSocket method
- onWebSocketOpen, OnWebSocketSession, onWebSocketConnect,
- onWebSocketMessage String Message, Session session), onWebSocketClose or following method WebSocket (maxTextMessageSize) when using Eclipse Jetty WebSocket implementations.

Drawing a Table to code for the sequences of sessions using WebSockets

- Column 3 may specify for each action, the authentication code or method (device
- platform ID, such as MAC address)
- Column 4 specifies the API inputs/parameters for initiating the action

Drawing a Table to code for the sequences of sessions using WebSockets

- Column 5 specifies the API outputs for the inputs. The outputs communicate to other end and initiate the execution of methods, callback functions, generate cookie or response.

Summary

We learnt

- Message exchanges between APIs, webAPIs, webService APIs, webClient and webServer using the REST APIs and WebSockets.
- Eclipse Jetty, Paho, and Ponte implementations provide the library functions for WebSockets and WebSocket APIs.

End of Lesson 18 on Prototyping Online Components for using the REST and WebSockets