

Mobile Programming

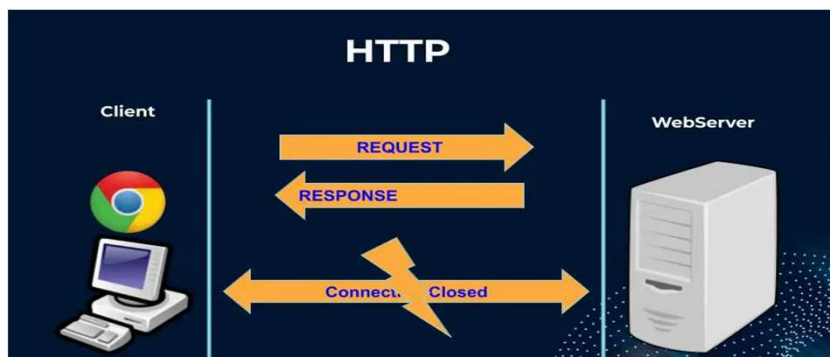
Chapter 6: WEBSOCKETS

Mobile programming

1

1. Introduction to WebSockets

WebSockets are a protocol for full-duplex communication channels over a single, long-lived TCP connection. They allow for real-time data transfer between a client (such as a web browser) and a server with low overheads and low latency.

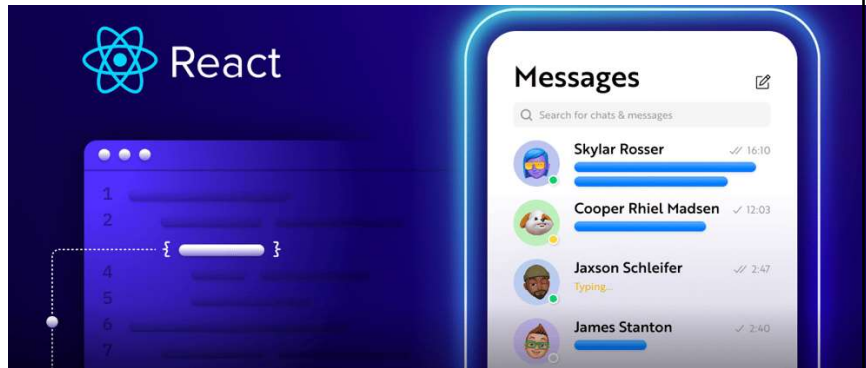


2

2

2. Use Cases

- Chat Applications
- Live News Updates
- Online Gaming
- Financial Tickers
- Collaborative Tools



Mobile programming

3

3

3. Advantages of WebSocket

- Real-time Communication
- Low Latency:
- Reduced Overhead
- Scalability
- Efficiency

Mobile programming

4

4

4. Disadvantages of WebSocket

- Complexity
- Compatibility
- Security Concerns
- Firewall and Proxy Issues
- Resource Management

Mobile programming

5

5

5. Server side implementations

•Node.js

- [Socket.IO](#)
- [WebSocket-Node](#)
- [ws](#)

•Java

- [Jetty](#)

•Ruby

- [EventMachine](#)

•Python

- [pywebsocket](#)
- [Tornado](#)

•Erlang

- [Shirasu](#)

•C++

- [libwebsockets](#)

•.NET

- [SuperWebSocket](#)

Mobile programming

6

6

5. Create the Node.js server

1. Create new directory
mkdir websockets
2. install ws, a WebSockets library for Node.js:
npm install ws
3. Open the websockets directory using your preferred text editor, like VSCode

```
[🐼 ~/repos/websockets (master|...)] tree -C -L 1
├── node_modules
├── package-lock.json
└── package.json
```

Mobile programming

7

7

5. Create the Node.js server

Open the Package.json file and add the following:

```
{
  "type": "module",
  "dependencies": {
    "ws": "^8.12.0"
  }
}
```

Mobile programming

8

8

5. Create the Node.js server

Create a index.js file with the content as below

```
import { WebSocketServer } from 'ws';

const wss = new WebSocketServer({ port: 8080 });

wss.on('connection', function connection(ws) {
  ws.on('message', function message(data) {
    console.log('received: %s', data);
  });

  ws.send('something');
});
```

Mobile programming

9

9

5. Create the Node.js server

Update libraries

npm update

Run server using Node JS

Node index.js

Mobile programming

10

10

5. Create native React apps using Websocket

Create websocket objects:

```
const websocket = new WebSocket('ws://my-websocket-server.com');
```

Handling connection:

```
websocket.onopen = (event) => {  
  console.log('WebSocket connection opened:', event);  
};
```

Mobile programming

11

11

5. Create native React apps using Websocket

Handling **incoming** messages:

```
websocket.onmessage = (event) => {  
  console.log('Received from server:', event.data);  
};
```

Sending messages to the server

```
websocket.send('Hello server');
```

Mobile programming

12

12

5. Create native React apps using Websocket

Handling connection error and closure:

```
websocket.onerror = (event) => {  
  console.log('WebSocket error:', event);  
};
```

```
websocket.onclose = (event) => {  
  console.log('WebSocket connection closed:', event);  
};
```

Closing the WebSocket connection

```
websocket.close();
```

Mobile programming

13

13

6. Example

Server:

```
const WebSocket = require('ws');
```

```
const server = new WebSocket.Server({ port: 8080 });
```

```
server.on('connection', ws => {  
  console.log('Client connected');
```

```
  ws.on('message', message => {  
    console.log('Received: ${message}');  
    // Gửi lại thông điệp tới client  
    ws.send(`Server received: ${message}`);  
  });
```

```
  ws.on('close', () => {  
    console.log('Client disconnected');
```

```
  });  
  console.log('WebSocket server is running on ws://localhost:8080');
```

14

14

6. Example

Client

```
import { useEffect, useState } from 'react';
import { View, Text, Button } from 'react-native';
export default function TestSocket() {
  const [data, setData] = useState('');
  const websocket = new WebSocket('ws://10.0.2.2:8080');
  websocket.onopen = function () {
    alert('Connect successful');
  }
  websocket.onerror = function (error) {
    console.log('Error: ' + JSON.stringify(error));
  }
  websocket.onmessage = function (e) {
    console.log('data response', e.data);
  }
}
```

Mobile programming

15

15

6. Example

Client

```
function SendData() {
  var data = {
    title: 'Tieu de',
    message: 'Hãy nhận lấy thông báo của tôi',
    picture: 'abc.jpg'
  }
  websocket.send(JSON.stringify(data));
}

function SendArrayData() {
  const array = new Float32Array(5);
  for (var i = 0; i < array.length; ++i) {
    array[i] = i / 2;
  }
  websocket.send(array);
}
```

Mobile programming

16

16

6. Example

Client

```
return (  
  <View><Text>abcd</Text>  
    <Button onPress={SendData} title='SendData'> </Button>  
    <Button onPress={SendArrayData} title='Send array data'> </Button></View>  
);  
}
```

Mobile programming

17

17



The slide features a dark blue background with a large white 'Q&A' text in the center. On the left, there is a photograph of a university building with a large tree in the foreground. In the top right corner, the EIU logo is displayed, consisting of a stylized globe icon followed by the text 'EIU' and 'TRƯỜNG ĐẠI HỌC QUỐC TẾ MIỀN ĐÔNG' and 'EASTERN INTERNATIONAL UNIVERSITY' below it. A small number '18' is visible in the bottom right corner of the slide area.

18