□□ WebSocket

Dieser Blogbeitrag wurde mit Unterstützung von ChatGPT-4o erstellt.

Einführung

Hallo zusammen, ich bin Li Zhiwei. Als Gründer und CTO der CodeReview-Plattform sowie ehemaliger Ingenieur bei LeanCloud, habe ich umfangreiche Erfahrungen mit WebSocket, insbesondere bei der Entwicklung von IM SDKs.

Die Bedeutung von WebSocket

WebSocket ist ein Protokoll, das einen vollständig duplexfähigen Kommunikationskanal über eine einzelne TCP-Verbindung bereitstellt. Es wird häufig in modernen Anwendungen eingesetzt, die Echtzeit-Interaktion erfordern, wie z.B. Instant Messaging, Echtzeit-Kommentare, Mehrspieler-Spiele, kollaborative Bearbeitung und Echtzeit-Aktienkurse.

Moderne Anwendungen von WebSocket

WebSocket wird in folgenden Bereichen häufig eingesetzt: - **Instant Messaging (IM)** - **Echtzeit-Kommentare** - **Mehrspieler-Spiele** - **Kollaboratives Bearbeiten** - **Echtzeit-Aktienkurse**

Die Entwicklung von WebSocket

Polling: Der Client fragt den Server häufig nach Updates an. **Long Polling:** Der Server hält die Anfrage offen, bis neue Informationen verfügbar sind. **HTTP-Zweikanalverbindung:** Erfordert mehrere Verbindungen zum Senden und Empfangen, und jede Anfrage enthält HTTP-Header. **Einzelne TCP-Verbindung (WebSocket):** Überwindet die Einschränkungen der HTTP-Zweikanalverbindung und bietet höhere Echtzeitfähigkeiten und geringere Latenz.

WebSocket auf iOS implementieren

Beliebte iOS WebSocket-Bibliotheken: - **SocketRocket□Objective-C□4910 Sterne□** - **Starscream□Swift□1714 Sterne□** - **SwiftWebSocket□Swift□435 Sterne□**

Verwendung von SRWebSocket

1. **Initialisierung und Verbindung:**

```
SRWebSocket *webSocket = [[SRWebSocket alloc] initWithURLRequest:[NSURLRequest requestWithURL:[NSUR
webSocket.delegate = self;
[webSocket open];
```

2. **Nachricht senden:**

```
[webSocket send:@"Hello, World!"];
```

3. **Nachrichten empfangen:** Implementieren Sie die `SRWebSocketDelegate`-Methoden, um eingehende Nachrichten und Ereignisse zu verarbeiten.

4. **Fehlerbehandlung und Ereignisbenachrichtigung:** Behandeln Sie Fehler angemessen und benachrichtigen Sie Benutzer über Verbindungsprobleme.

Detaillierte Erklärung des WebSocket-Protokolls

WebSocket läuft auf TCP und führt mehrere Verbesserungen ein: - **Sicherheitsmodell:** Ein browserbasiertes Ursprungs-Sicherheitsüberprüfungsmodell wurde hinzugefügt. - **Adress- und Protokollbenennung:** Unterstützung für mehrere Dienste auf einem einzelnen Port und mehrere Domänen auf einer einzelnen IP-Adresse. - **Rahmenmechanismus:** TCP wurde durch einen rahmenähnlichen Mechanismus ähnlich IP-Paketen verbessert, ohne Längenbeschränkungen. - **Schließhandshake:** Stellt sicher, dass die Verbindung sauber geschlossen wird.

Das Kernkonzept des WebSocket-Protokolls

1. Handshake: Der WebSocket-Handshake verwendet den HTTP-Upgrade-Mechanismus: - **Client-Anfrage:** `http GET /chat HTTP/1.1 Host: server.example.com Upgrade: websocket Connection: Upgrade Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ== Origin: http://example.com Sec-WebSocket-Protocol: chat, superchat Sec-WebSocket-Version: 13`

- **Server-Antwort:** `http HTTP/1.1 101 Switching Protocols Upgrade: websocket Connection: Upgrade Sec-WebSocket-Accept: s3pPLMBiTxaQ9kYGzzhZRbK+xOo= Sec-WebSocket-Prot chat`

2. Datenübertragung: WebSocket-Frames können UTF-8-Text, Binärdaten und Steuerframes wie Schließen, Ping und Pong enthalten.

3. Sicherheit: Der Browser fügt automatisch den Origin-Header hinzu, der von anderen Clients nicht gefälscht werden kann.

WebSocket-URI

- **ws-URI:** `ws://host:port/path?query`
- **wss-URI:** `wss://host:port/path?query`

WebSocket-Frame-Protokoll

Rahmenstruktur: - **FIN (1 Bit):** Gibt an, dass dies das letzte Fragment der Nachricht ist. - **RSV1, RSV2, RSV3 (je 1 Bit):** Für zukünftige Verwendung reserviert. - **Opcode (4 Bits):** Definiert, wie die Nutzdaten interpretiert werden sollen. - `0x0`: Fortsetzungsrahmen - `0x1`: Textrahmen - `0x2`: Binärrahmen - `0x8`: Verbindungsschließen - `0x9`: Ping - `0xA`: Pong - **Mask (1 Bit):** Gibt an, ob die Nutzdaten maskiert sind. - **Nutzlastlänge (7 Bits):** Die Länge der Nutzdaten.

Mask Key: Wird verwendet, um Man-in-the-Middle-Angriffe zu verhindern, indem die Frames des Clients maskiert werden.

Handshake beenden

Schließrahmen (Close Frame): - Kann einen Körper enthalten, der den Grund für das Schließen angibt. - Beide Seiten müssen einen Schließrahmen senden und darauf antworten.

Beispiel

Beispiel 1: Einzelner ungemaskierter Textnachrichten-Frame

```
0x81 0x05 0x48 0x65 0x6c 0x6c 0x6f
```

Enthält “Hello”

Beispiel 2: Maskierte Textnachricht in einem Einzelrahmen

```
0x81 0x85 0x37 0xfa 0x21 0x3d 0x7f 0x9f 0x4d 0x51 0x58
```

Enthält “Hello” mit einem Maskierungsschlüssel.

Beispiel 3: Fragmentierte unverschlüsselte Textnachricht

```
0x01 0x03 0x48 0x65 0x6c
0x80 0x02 0x6c 0x6f
```

Die Fragmente enthalten die beiden Frames “Hel” und “lo”.

Fortgeschrittene Themen

Masking und Demasking: - Masking wird verwendet, um Man-in-the-Middle-Angriffe zu verhindern. - Jeder Frame vom Client muss maskiert werden. - Der Maskierungsschlüssel für jeden Frame wird zufällig ausgewählt.

Fragmentierung: - Wird verwendet, um Daten unbekannter Länge zu senden. - Fragmentierte Nachrichten beginnen mit einem Frame, bei dem FIN auf 0 gesetzt ist, und enden mit einem Frame, bei dem FIN auf 1 gesetzt ist.

Steuerungsrahmen: - Steuerungsrahmen (wie Schließen, Ping und Pong) haben spezifische Opcodes. - Diese Rahmen werden verwendet, um den Zustand der WebSocket-Verbindung zu verwalten.

Erweiterbarkeit

Erweiterte Daten können vor den Anwendungsdaten im Nachrichtenkörper platziert werden: - Reservierte Bits können verwendet werden, um jeden Frame zu steuern. - Einige Opcodes sind für zukünftige Definitionen reserviert. - Falls mehr Opcodes benötigt werden, können die reservierten Bits verwendet werden.

Senden: - Es muss sichergestellt werden, dass die Verbindung im Zustand OPEN ist. - Die Daten werden in Frames verpackt, und bei zu großen Daten kann eine Aufteilung in mehrere Frames erfolgen. - Der Wert des ersten Frames muss korrekt sein, um dem Empfänger den Datentyp (Text oder Binär) mitzuteilen. - Das FIN-Bit des letzten Frames muss auf 1 gesetzt werden.

Schließen des Handshakes: - Beide Seiten können einen Schließrahmen senden. - Nach dem Senden eines Schließrahmens werden keine weiteren Daten gesendet. - Nach dem Empfang eines Schließrahmens werden alle danach empfangenen Daten verworfen.

Verbindung schließen: - Schließen der WebSocket-Verbindung, was das Schließen der darunterliegenden TCP-Verbindung bedeutet. - Nach dem Senden oder Empfangen eines Schließrahmens (Close Frame) befindet sich die WebSocket-Verbindung im Zustand “wird geschlossen”. - Wenn die darunterliegende TCP-Verbindung geschlossen ist, befindet sich die WebSocket-Verbindung im Zustand “geschlossen”.

Referenzen

- WebSocket RFC: RFC6455
- Zhihu□WebSocket □□□□□□□: Zhihu □□
- SocketRocket: GitHub □□

Danksagung

Vielen Dank für Ihre Aufmerksamkeit. Wenn Sie weitere Fragen oder Diskussionen haben, können Sie mich gerne auf GitHub oder Weibo kontaktieren.