socket.io is a real-time library for building scalable, high-performance applications like chat applications, multiplayer games, collaborative editors, etc. It provides a simple API for connecting JavaScript and Node.js applications to web sockets.

### Socket.io is Multiple Chat Rooms

Socket.io is a popular real-time library that allows developers to build real-time chat applications. It provides a simple API for connecting JavaScript and Node.js applications to web sockets, enabling developers to build scalable, high-performance chat applications.

#### Socket.io Multiple Chat Rooms

Node.js version: 14.17.0

**Socket.io is Multiple Chat Rooms**

Socket.io is a real-time library for building scalable, high-performance applications like chat applications, multiplayer games, collaborative editors, etc. It provides a simple API for connecting JavaScript and Node.js applications to web sockets.

### How to Create Multiple Chat Rooms

To create multiple chat rooms using Socket.io, you can use the `io.of()` method to create a new instance of the socket.io namespace. You can then use this namespace to create multiple chat rooms.

```javascript
const { Server } = require('socket.io');
const app = new Server(8000);
const chat = io.of('/chat');

chat.on('connection', (socket) => {
  socket.on('newMessage', (message) => {
    chat.emit('newMessage', message);
  });

  socket.on('JoinedRoom', (roomId) => {
    chat.join(socket, roomId);
    socket.broadcast.to(roomId).emit('RoomJoined', roomId);
  });

  socket.on('leaveRoom', (roomId) => {
    chat.leave(socket, roomId);
    socket.broadcast.to(roomId).emit('RoomLeft', roomId);
  });
});
```

In this example, we create a new instance of the socket.io namespace called `chat`. We then use the `chat` namespace to create multiple chat rooms by calling the `chat.join(socket, roomId)` method. We also broadcast a message to all clients in the room using the `chat.emit()` method.

### Handling Chat Messages

When a user sends a message, you can listen for the `newMessage` event in the chat namespace to receive the message. Then, you can broadcast the message to all clients in the room.

```javascript
chat.on('newMessage', (message) => {
  chat.broadcast.to(roomId).emit('newMessage', message);
});
```

### Adding Joining and Leaving Room Support

You can add functionality for users to join and leave chat rooms. To do this, you can listen for the `JoinedRoom` and `leaveRoom` events in the chat namespace.

```javascript
chat.on('JoinedRoom', (roomId) => {
  // handle user joining a room
});

chat.on('leaveRoom', (roomId) => {
  // handle user leaving a room
});
```

### Security Considerations

When building chat applications, security is a critical concern. You should always use secure connections (e.g., HTTPS) and ensure that your chat application is properly secured.

---

**See More**

**Build a ChatApp with: (Nodejs,Socketio, Expressjs ,MongoDB)**

**Io protocol supports**

**socket io multiple chat rooms. Please bear with me. I originally followed this tutorial to get my node.**

**It will create multiple**

**ChatServer with Multiple Rooms**

**socket.**

**The ws or websocket protocol is an upgrade from...**

**Build Multiple ChatRooms with Node.js, Socket.IO and MongoDB**

**28.01.2021 by Mikajind. This multi-room chat example is part 2 in the node.**

**Thanks for the article, really...**

**from the TURN...**

**chat, and webinars across mobile, desktop, and room systems.**

**2015 · Edit: A nice implementation**

**socket io multiple chat rooms. Please bear with me. I originally followed this tutorial to get my node.**

**It will create multiple**

**ChatServer with Multiple Rooms**

**socket.**

**The ws or websocket protocol is an upgrade from...**

**Build Multiple ChatRooms with Node.js, Socket.IO and MongoDB**

**28.01.2021 by Mikajind. This multi-room chat example is part 2 in the node.**

**Thanks for the article, really...**

**from the TURN...**

**chat, and webinars across mobile, desktop, and room systems.**

**2015 · Edit: A nice implementation**

**socket io multiple chat rooms. Please bear with me. I originally followed this tutorial to get my node.**

**It will create multiple**

**ChatServer with Multiple Rooms**

**socket.**

**The ws or websocket protocol is an upgrade from...**

**Build Multiple ChatRooms with Node.js, Socket.IO and MongoDB**

**28.01.2021 by Mikajind. This multi-room chat example is part 2 in the node.**

**Thanks for the article, really...**

**from the TURN...**

**chat, and webinars across mobile, desktop, and room systems.**

**2015 · Edit: A nice implementation**

**socket io multiple chat rooms. Please bear with me. I originally followed this tutorial to get my node.**

**It will create multiple**

**ChatServer with Multiple Rooms**

**socket.**

**The ws or websocket protocol is an upgrade from...**

**Build Multiple ChatRooms with Node.js, Socket.IO and MongoDB**

**28.01.2021 by Mikajind. This multi-room chat example is part 2 in the node.**

**Thanks for the article, really...**

**from the TURN...**

**chat, and webinars across mobile, desktop, and room systems.**

**2015 · Edit: A nice implementation**

**socket io multiple chat rooms. Please bear with me. I originally followed this tutorial to get my node.**

**It will create multiple**

**ChatServer with Multiple Rooms**

**socket.**

**The ws or websocket protocol is an upgrade from...**

**Build Multiple ChatRooms with Node.js, Socket.IO and MongoDB**

**28.01.2021 by Mikajind. This multi-room chat example is part 2 in the node.**

**Thanks for the article, really...**