Socket.io

Котович Дмитрий

Лаптев Никита

What is it?

JavaScript-библиотека для веб-приложений и обмена данными в реальном времени. Состоит из двух частей: клиентской, которая запускается в браузере и серверной для node.js. Является событийно-ориентированным API.

Using Socket.IO

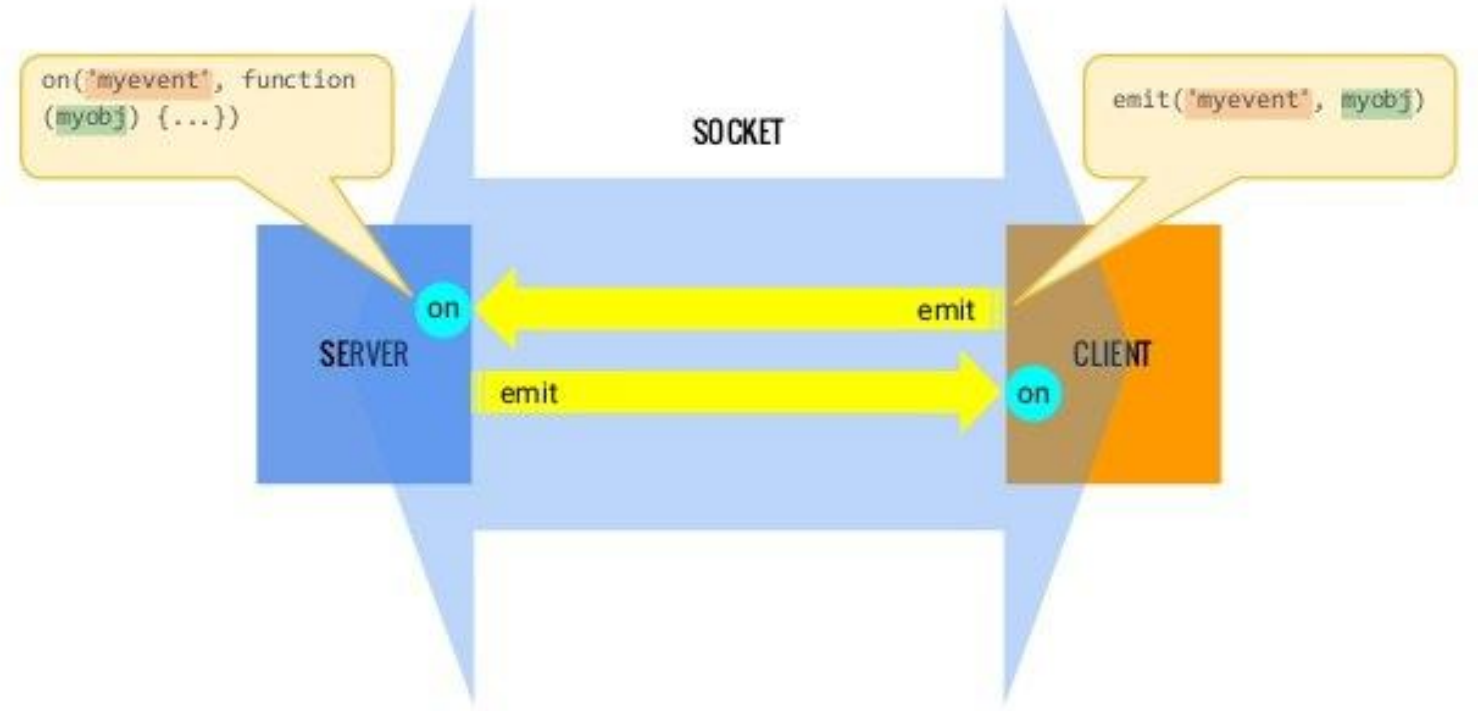
Ответ: real-time applications

Who use?

1. Microsoft Office
2. Yammer
3. Zendesk
4. Trello

{
“Type”: “event-oriented”,
“Author”: “Guillermo Rauch”,
“Version”: “v2.0.3”,
“work”: “npm install --save socket.io”
“url”: “github.com/socketio/socket.io”,
“githubStars” : 37777
}

Взаимодействие client-server



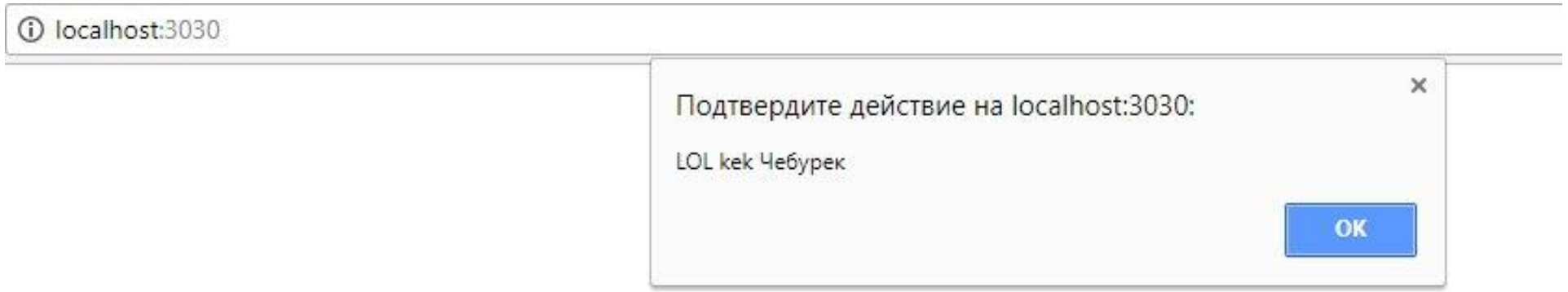
Server example

```
const app = require('express')();
const http = require('http').Server(app);
const io = require('socket.io')(http);
app.get('/', (req, res) => {
    res.sendFile(__dirname + '/index.html');
});
io.on('connection', (socket) => {
    socket.emit("hello", "LOL kek Чебурек");
});
http.listen(3030, '127.0.0.1', () => {
    console.log('Lol kek server is running');
});
```

Client example

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Socket.io</title>
    <script src="/socket.io/socket.io.js"></script>
</head>
<body>
    <script>
        const socket = io();
        socket.on("hello", (serverData) => {
            alert(serverData);
        });
    </script>
</body>
</html>
```

Result


localhost:3030
Подтвердите действие на localhost:3030:
LOL kek Чебурек
OK

Working in browsers

1. WebSocket
2. Adobe Flash Socket
3. AJAX long polling
4. AJAX multipart streaming
5. Forever Iframe
6. JSONP Polling

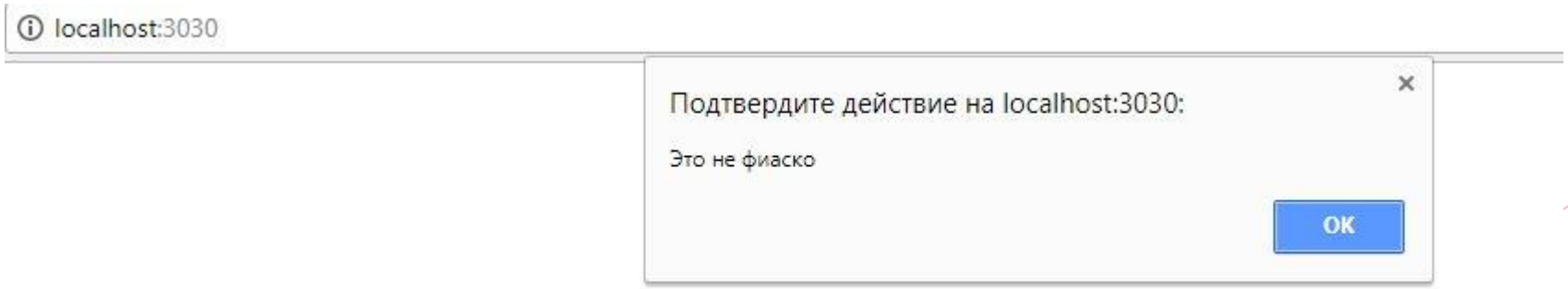
Main methods

- sockets – выбор всех подключённых клиентов
- socket.send(TEXT) – отправка сообщения TEXT
- socket.json.send({}) – отправка JSON-сообщения
- socket.broadcast.send – широковещательное сообщение
- socket.emit(EVENT, ANY) – отправка сообщения по событию
- socket.on(EVENT, CALLBACK) – вызов метода в ответ на событие

send()

```
const app = require('express')();
const http = require('http').Server(app);
const io = require('socket.io')(http);
app.get('/', (req, res) => {
    res.sendFile(__dirname + '/index.html');
});
io.on('connection', (socket) => {
    socket.send("Это не фиаско");
});
http.listen(3030, '127.0.0.1', () => {
    console.log('Lol kek server is running');
});
```

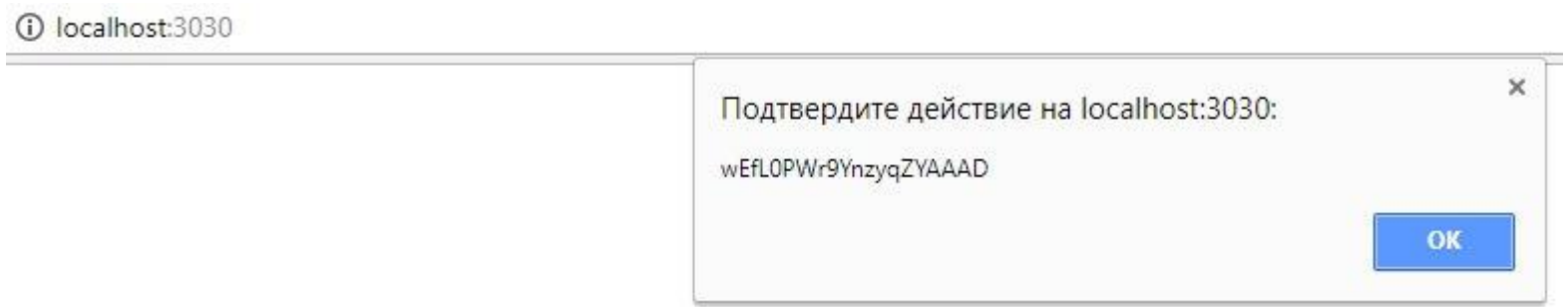
```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Socket.io</title>
    <script src="/socket.io/socket.io.js"></script>
</head>
<body>
    <script>
        const socket = io();
        socket.on("message", (serverData) => {
            alert(serverData);
        });
    </script>
</body>
</html>
```



json.send({})

```
const app = require('express')();
const http = require('http').Server(app);
const io = require('socket.io')(http);
app.get('/', (req, res) => {
    res.sendFile(__dirname + '/index.html');
});
io.on('connection', (socket) => {
    socket.json.send({'id': socket.id});
});
http.listen(3030, '127.0.0.1', () => {
    console.log('Lol kek server is running');
});
```

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Socket.io</title>
    <script src="/socket.io/socket.io.js"></script>
</head>
<body>
    <script>
        const socket = io();
        socket.on('message', (data) => { alert (data.id); });
    </script>
</body>
</html>
```



emit(EVENT,ANY)

Server side:

1. Connect
2. Reconnect
3. Disconnect
4. Ping
5. Leave
6. Message
7. Join

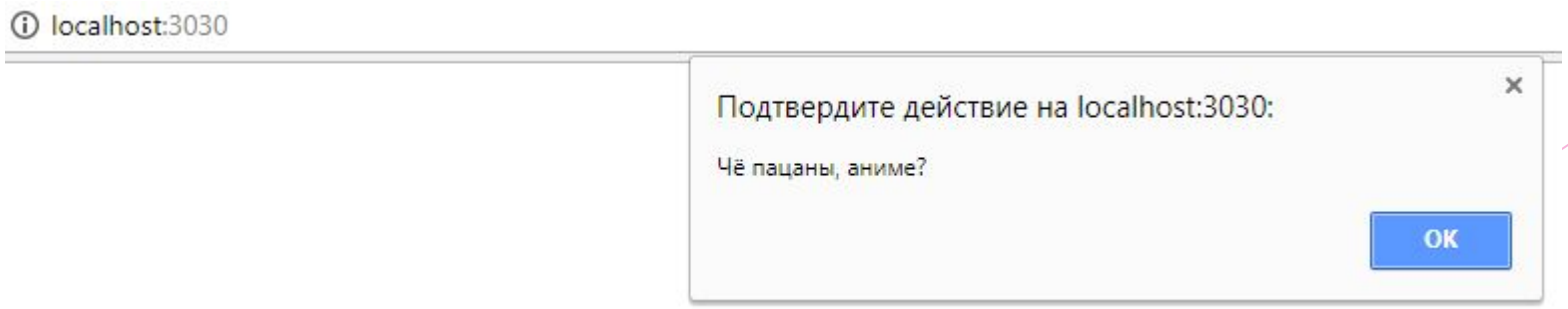
Client side:

1. Connect
2. Connect_error
3. Connect_timeout
4. Reconnect

Namespaces

```
const app = require('express')();
const http = require('http').Server(app);
const io = require('socket.io')(http);
app.get('/', (req, res) => {
    res.sendFile(__dirname + '/index.html');
});
const nsp = io.of('/my-namespace');
nsp.on('connection', (socket) => {
   nsp.emit('hi', 'Чё пацаны, аниме?');
});
http.listen(3030, '127.0.0.1', () => {
    console.log('Lol kek server is running');
});
```

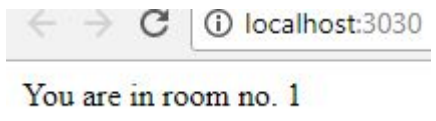
```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Socket.io</title>
    <script src="/socket.io/socket.io.js"></script>
</head>
<body>
    <script>
        const socket = io('/my-namespace');
        socket.on('hi', (data) => { alert (data); });
    </script>
</body>
</html>
```

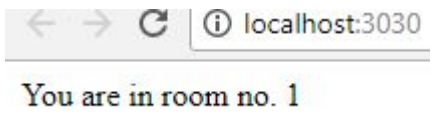


Rooms

```
const app = require('express')();
const http = require('http').Server(app);
const io = require('socket.io')(http);
app.get('/', (req, res) => {
    res.sendFile(__dirname + '/index.html');
});
let roomno = 1;
io.on('connection', (socket)=> {
   if(io.nsps['/'].adapter.rooms["room-"+roomno]
      && io.nsps['/'].adapter.rooms["room-"+roomno].length > 1) roomno++;
   socket.join("room-"+roomno);
   io.sockets.in("room-"+roomno).emit('connectToRoom', "You are in room no. "+roomno);
})
http.listen(3030, '127.0.0.1', () => {
    console.log('Lol kek server is running');
});
```

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Socket.io</title>
    <script src="/socket.io/socket.io.js"></script>
</head>
<body>
    <script>
        const socket = io();
        socket.on('connectToRoom', (data) =>
        {
            document.body.innerHTML = data;
        });
    </script>
</body>
</html>
```





Спасибо за внимание!