



REALTIME WEB APPS

Fingoweb
PSDorganizer


SLAWOMIR.WILUSZ@FINGOWEB.COM - FINGOWEB.COM - PSDORGANIZER.COM - @2017

POLLING

- `<meta http-equiv="refresh" content="5">`
- `window.location + setTimeout`
- AJAX
 - Short polling
 - Long polling

PUSH

- server-sent event

Server-sent events  - LS

Global88.27% + 0.02% = 88.29%

Method of continuously sending data from a server to the browser, rather than repeatedly requesting it (EventSource interface, used to fall under HTML5)

Current alignedUsage relativeDate relativeShow all

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
			49						
			55			9.3		4.4	
	14	51	56	10	43	10.2		4.4.4	
11	15	52	57	10.1	44	10.3	all	56	57
		53	58	TP	45				
		54	59		46				
		55	60						

Notes

Known issues (4)

Resources (7)


Feedback

MS Edge status: Under Consideration

WEBSOCKETS

Advantages of Websockets over SSE:

- Real time, two directional communication.
- Native support in more browsers

Web Sockets  - LS

Global 93.73% + 0.26% = 93.98%

unprefixed: 93.73% + 0.21% = 93.94%

Bidirectional communication technology for web apps

Current aligned Usage relative Date relative Show all

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
			49						
			55			9.3		4.4	
	14	51	56	10	43	10.2		4.4.4	
11	15	52	57	10.1	44	10.3	all	56	57
		53	58	TP	45				
		54	59		46				
		55	60						

Notes Known issues (1) Resources (10) Feedback

Reported to be supported in some Android 4.x browsers, including Sony Xperia S, Sony TX and HTC.

SERVER-SIDE – IN MEMORY

- socket.io

Socket.IO is a JavaScript library for realtime web applications. It enables realtime, bi-directional communication between web clients and servers. It has two parts: a client-side library that runs in the browser, and a server-side library for Node.js. Both components have a nearly identical API.

SERVER-SIDE – DB CHANGE FEED

- Oplog / Binlog
 - MongoDB
 - MySql
- Triggers
 - PostgreSQL – LISTEN/NOTIFY
- Dedicated solution
 - RethinkDB – „changes()”
 - CouchDB – „_changes”

PUTTING THE PIECES TOGETHER

- React
- NodeJS
- FeathersJS + RxJS + Socket.io
- RethinkDB

DEMO – SERVER SIDE



```
package.json x index.html x app.js x client.js x
const feathers = require('feathers');
const socketio = require('feathers-socketio');
const handler = require('feathers-errors/handler');
const r = require('rethinkdbdash')({
  db: 'realtimeapp'
});
const rethinkdb = require('feathers-rethinkdb');
const app = feathers();

app.configure(socketio());

app.use('/comments', rethinkdb({
  Model: r,
  name: 'comments'
}));

app.use('/', feathers.static(__dirname));
app.use(handler());
app.listen(3001);
```


DEMO – CLIENT SIDE

```
package.json x index.html x app.js x client.js x
1 import io from 'socket.io-client';
2 import feathers from 'feathers/client';
3 import socketio from 'feathers-socketio/client';
4 import React from 'react';
5 import ReactDOM from 'react-dom';
6 import rx from 'feathers-reactive';
7 import RxJS from 'rxjs';
8
9 const socket = io();
10 const app = feathers()
11   .configure(socketio(socket))
12   .configure(rx(RxJS));
13
14 const commentService = app.service('comments');
15
16 class CommentsList extends React.Component {
17   componentDidMount() {
18     commentService.find({ query: {
19       $sort: {
20         created: 1
21       }
22     }})
23     .subscribe(comments => this.setState({comments}));
24   }
25
26   remove(item) {
27     commentService.remove(item.id);
28     return false;
29   }
30
31   render() {
32     if (this.state && this.state.comments) {
33       return <div>
34         {this.state.comments.map(comment =>
35           <div key={comment.id}>
36             <div>{comment.text}</div>
37             <pre>Created: {new Date(comment.created).toISOString()} <a href="#" onClick={this.remove.bind(this, comment)}>x</a><br/></pre>
38             <hr/>
39           </div>
40         )}
41         <pre>Rendered: {new Date().toISOString()}</pre>
42       </div>;
43     }
44     return null;
45   }
46 }
47
```

DEMO – CLIENT SIDE 2

```
package.json x index.html x app.js x client.js x
48
49 class CommentForm extends React.Component {
50   constructor(props) {
51     super(props);
52     this.state = {
53       text: '',
54       created: null
55     };
56   }
57
58   updateProperty(prop, ev) {
59     this.setState({[prop]: ev.target.value});
60   }
61
62   submit(ev) {
63     ev.preventDefault();
64
65     commentService.create({
66       text: this.state.text,
67       created: Date.now()
68     }).then(() => this.setState({
69       text: '',
70       created: null
71     }));
72   }
73
74   render() {
75     return <form onSubmit={this.submit.bind(this)}>
76       <div>
77         <div>Comment:</div>
78         <div>
79           <input type="text" onChange={this.updateProperty.bind(this, 'text')} value={this.state.text}/>
80         </div>
81       </div>
82       <div>
83         <button type="submit">Add</button>
84       </div>
85     </form>;
86   }
87 }
88
89 ReactDOM.render(<div>
90   <CommentsList />
91   <CommentForm />
92 </div>, document.getElementById('app'));
93
```

DEMO – PSDORGANIZER.COM

PSDorganizer

THE END

THANK YOU

Fingoweb

PSDorganizer