Question
Sean Connelly · Apr 19, 2017

WebSockets vs Long Polling vs Short Polling?

WebSockets look to be supported reasonably well in Cache. I have yet to use them in production so I am wondering how well it has worked for other developers.

In particular what happens when the browser does not support WebSockets, or when a firewall blocks the connection.

Have you had to write your own long polling fall-back?

I've read the documentation and found this interesting article...

https://community.intersystems.com/post/asynchronous-websockets-quick-tutorial

but no mention of long polling fall-backs that I can see.

Having tinkered around with sockjs and socket.io (with node) there is a realisation that these libraries require a great deal of fringe code to deal with a vast number of issues, such as fall-backs, disconnects and fringe errors, that I worry about using WebSockets in production.

Interestingly I have also read that facebook and twitter have used (and may still use) long polling instead of websockets because of various issues they have experienced. Considering the millions of users they have constantly connected, it makes me wonder if the fuss with long polling has more to do with how its implemented. If they can make it work for millions on technology x, then why not for Caché.

In particular, if the applications that I develop are intranet wide and not internet wide, then should I save the technical headache and just implement long polling instead for a smaller number of users. If I was using node.js then I would probably lean towards this decision based on my numbers of users. Except if I am using Caché only I wonder how I would implement this without each users connection creating a hung process until something happens.

I could write an intermediary message broker in node.js, but ideally I want a solution that is Caché only.

In the past I have developed short polling solutions in Caché. Requests that are known to have async operations are queued and the client is asked to periodically check back for the response. In benchmarks this is still able to handle a colossal number of messages, way more than my users ever need. It's simplicity is a grade trade off for the small lag that my users probably don't notice. But, removing that lag would still be good.

Any experience or advice would be appreciated. Is short polling working just fine for you? Does anyone have a clever Caché only long poll solution they can share that does not hang a process? Are you more than happy with websockets alone, or have you had to implement fall-backs as well?

Sean.

#Node.js #Tutorial #CSP #Caché