Why does SIgnalR prefer Forever Frames over polling?

I am learning to use SignalR and so far I had success in doing so. I can implement Hubs, I can implement business logic, I can invoke client-side functions from server for whom I want, I can invoke server-side methods from client-side, this stuff is great. The thing puzzling me is the theory.

In fact, I gathered info from this video. SignalR is using WebSockets, which provide a full duplex channel over a single TCP connection. If no WebSockets are available, then the fallback protocol will be EventSource. If that is unavailable, then a Forever Frame will be used. If that is unavailable, then Long Polling will be used. It is rather strange for me that a very hacky solution, like a forever frame is preferred over the older convention and I am interested about the rationale behind SignalR's decision to have forever frames as the third option and polling as fourth option.

I have tried to find out the answer to this question and I found out that it is rumored that there is a 3x max latency time in the case of long polling compared to forever frames. Is this a fact and if so, is it a fact for all browsers, or for a subset?

asked Sep 16 '16 at 1:06

Lajos Arpad
27.2k 18 61 116

1 Answer

foreverFrame works a bit like serverSentEvents - there is one long running http request which the server never terminates but uses to push data to the client. longPolling works differently - there is a poll http request which is being closed by the server each time there is data for the client to read (or a timeout expires). If the client wants to read more data it needs to open a new http request which again will be closed by the server as soon as there is any data for the client. In other words, in case of foreverFrame the server is pushing data to the client using an already established channel while in case of longPolling the client is continuously creating http requests to get the data from the server.

answered Sep 16 '16 at 2:27

Pawel
27.4k 4 55 90

Understood and thank you for your answer, which already deserved an upvote. However, I wonder whether you can factually confirm that the 3x latency due to the need of opening an HTTP channel each time the Poller sends data actually exists. – Lajos Arpad Sep 16 '16 at 7:30

Sending data works the same way - a new http request is created in both cases so I would say there is no difference in latency between foreverFrame and longPolling when sending data. – Pawel Sep 16 '16 at 14:00

Is there a latency at receiving time? – Lajos Arpad Sep 16 '16 at 14:15

http request each time while foreverFrame doesn't have to do that because it is using one request for receiving all data. – Pawel Sep 16 '16 at 14:26