Long polling

long, polling, websocket, sse

What is Long polling

Long polling solves situations when the user is forced to periodically click on **refresh of the table to see the current data**. Long polling ensures page **refresh automatically, without user interaction**.

The easiest way to achieve automatic renewal is to periodically polling the client. The period of these requests must be as short as possible (seconds). This implies a major disadvantage of **short polling** and this is a large number of requests to the server.

Long polling works differently:

- The **client creates a request** to the server to be informed about the changes. This request is not immediately terminated, but is held until a change occurs. - If a change occurs, the **request is terminated** with a particular status. - **The client** can decide whether to refresh. - Immediately after the request is resolved, the **client creates a new request** to the server and the situation repeats.

Unfortunately, the time the request is held cannot be unlimited in the real world, for example because of the time constraint on the Appache server (60s). For this reason, the maximum length of one long polling request is limited to **30 seconds**. After this time, the request expires and the client makes a new request.

While holding a long polling request, it is periodically evaluated whether the entity has changed. This typically means that every **2 seconds**, all deferred requests are **evaluated for any change**. To correctly detect that a change has occurred, it is necessary to maintain certain metadata from the previous run. These metadata are the time point of the last entity change (**change / create detection**) and the last known number of entities (**delete detection**). These metadata are stored in memory as a subscriber map, where the key is the entity identifier.

To ensure that the subscriber map was not **too large**, a periodic release mechanism was implemented.

Each time a change is verified, each subscriber is marked with a **timestamp when the last verification occurred**. If the user navigates to another page, there will be no additional long polling requests. Thus, the deferred requests will expire and the time stamp will not be renewed at the customer. Every two hours, the `LongPollingManager.clearUnUseSubscribers` method is automatically scheduled to **remove any subscribers that have not been used in the last hour**.

Long polling vs. Websocket

The biggest disadvantage of Websockets is the use of its own **non-HTTP protocol**. **These ports can often be blocked** by customers and therefore Websockets **cannot be used**.

Where is Long polling used in IdM

For now is long polling implemented these agendas:

- **Identity** controller - for check changes on a role-requests.
- **System** controller - for check changes on a synchronizations.

See more

Do you really need WebSockets?

Using SSE Instead Of WebSockets For Unidirectional Data Flow Over HTTP/2

- `LongPollingManager`
- `IdmIdentityController`
- `SysSystemController`