Long and Short Polling

By Paige Schaefer

What does polling mean?

Polling is a technique by which the client asks the server for new data regularly

Client-Server Architecture

- A client makes an HTTP request requesting a web page from a server
- The server calculates the response
- The server sends the response to the client

Short Polling

- Client makes a request to the server
- Server can respond in two ways:
  - It sends an empty response
  - It sends data object in its body
- As soon as a client receives the response from the server, it will wait for a couple of seconds and repeat the above process

Long Polling

- Client makes a request to the server
- Server can respond in two ways:
  - It if has some new data available, it can respond right away
  - It if doesn’t have anything new data, it will keep that connection open for a period of time and when it receives new data it will respond back with updated data

```
00:00:00 C-> Is the cake ready?
00:00:01 S-> No, wait.
00:00:01 C-> Is the cake ready?
00:00:02 S-> No, wait.
00:00:02 C-> Is the cake ready?
00:00:03 S-> Yeah. Have some lad.
00:00:03 C-> Is the other cake ready? ..
```

```
12:00 00:00:00 C-> Is the cake ready?
12:00 00:00:03 S-> Yeah.Hame some lad.
12:00 00:00:03 C-> Is the cake ready?
```

Introduction

- Techniques to retrieve data from the server
- Client-Server Architecture
  - A client makes an HTTP request requestions a web page from a server
  - The server calculates the response
  - The server sends the response to the client
- Polling a technique by which the client asking the server the new data regularly

Short Polling

- Short polling is an AJAX-based timer that calls at fixed delays
- Short Polling Technique
  - Client makes a request to the server
  - Server can respond in two ways:
    - It sends an empty response
    - It sends data object in its body (JSON object)
  - As soon as a client receives the response from the server, it will wait for a couple of seconds and repeat the above process
- Some challenges in short-polling
  - Making repeated requests to the server wastes resources as each new incoming connection must be established, the HTTP headers must be passed a query for new data must be performed, and a response (usually with no new data to offer) must be generated and delivered. The connection must be closed and any resources cleaned up.

Long Polling

- Long Polling Technique
  - Client makes a request to the server
  - Server can respond in two ways:
    - It if has some new data available, it can response right away
    - It if doesn't have anything new data, it will keep that connection open for a period of time and when it receives new data it will respond back with updated data
- Client continuously asks the server for new information using regular HTTP requestions & the server stalls its answer when it has nothing new to report
- As long as the client makes sure it constantly has a polling requestion open, it will receive information from the server quickly after it becomes available
- To prevent connections from timing out (being aborted because of a lack of activity), long polling techniques usually set a maximum time for each request, after which the server will respond anyway, even though it has nothing to repeat, after which the client will start a new request.
- Some challenges in long-polling
  - Message ordering and delivery guarantees: Message ordering cannot be guaranteed if the same client opens multiple connections to the server.
  - If the client was not able to receive the message then there will be possible message loss.
  - Performance and scaling
  - Device support and fallbacks

Example Short Polling

```
00:00:00 C-> Is the cake ready?
00:00:01 S-> No, wait.
00:00:01 C-> Is the cake ready?
00:00:02 S-> No, wait.
00:00:02 C-> Is the cake ready?
00:00:03 S-> Yeah. Have some lad.
00:00:03 C-> Is the other cake ready? ..
```

Send a request to the server, get an instant answer. But: this costs a lot of requests.

Example Long Polling

```
12:00 00:00:00 C-> Is the cake ready?
12:00 00:00:03 S-> Yeah.Hame some lad.
12:00 00:00:03 C-> Is the cake ready?
```

Send a request to the server, keep the connection open, get an answer when there's "data" for you.
This will cost you only one request (per user) but the request keeps a permanent connection between client and server up
Traffic is smaller but you eat up your recourses faster (rather you block your resources)
Potentially in short poll you used more transfer, but during those 3s you take 1.5s of processing time
In case for long poll the same resources were used all the time

Website:
short polling: http://165.106.10.170:30009/s/short.html
long polling: http://165.106.10.170:30009/s/long.html

Both pages simply display a counter that is incremented every time you hit the page

http://165.106.10.170:30009/inc

Sources:
https://anuradha.hashnode.dev/short-polling-vs-long-polling-vs-web-sockets

https://qiuzhihui.gitbooks.io/r-book/content/system-design/short-polling-v-long-polling-vs-websocket.html