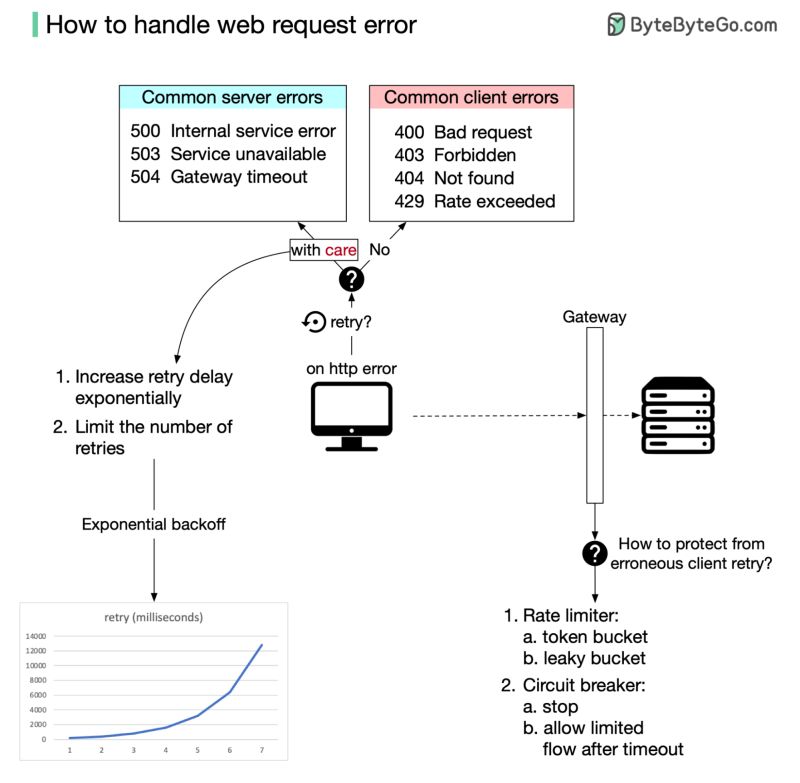
How do we properly deal with HTTP errors on the browser side? And how do we handle them correctly on the server side when the client side is at fault?

  
From the browser's point of view, the easiest thing to do is to try again and hope the error just goes away. This is a good idea in a distributed network, but we also have to be very careful not to make things worse. Here’s two general rules:  
  
1. For 4XX http error code, do not retry.  
2. For 5XX http error code, try again carefully.  
  
So which things should we do carefully in the browser? We definitely should not overwhelm the server with retried requests. An algorithm named exponential backoff might be able to help. It controls two things:  
  
1. The latency between two retries. The latency will increase exponentially.  
2. The number of retries is usually capped.  
  
Will all browsers handle their retry logic in a graceful way? Most likely not. So the server has to take care of its own safety. A common way to control the flow of HTTP requests is to set up a flow control gateway in front of the server. This provides two useful tools:  
  
1. Rate limiter, which limits how often a request can be made. It has two slightly different choices; the token bucket and the leaky bucket.  
2. Circuit breaker. This will stop the HTTP flow immediately when the error threshold is exceeded. After a set amount of time, it will only let a limited amount of HTTP traffic through. If everything works well, it will slowly let all HTTP traffic through.  
  
We should be able to handle intermittent errors effectively with exponential backoff in the browser and with a flow control gateway on the server side. Any remaining issues are real errors that need to be fixed carefully.  
  
Over to you: Both token bucket and leaky bucket can be used for rate limiting. How do you know which one to pick?  
  
—  
If you enjoy these posts, you might like my ByteByteGo 𝐒𝐲𝐬𝐭𝐞𝐦 𝐃𝐞𝐬𝐢𝐠𝐧 𝐧𝐞𝐰𝐬𝐥𝐞𝐭𝐭𝐞𝐫. Subscribe here: <https://bit.ly/3FEGliw>  
   
[#systemdesign](https://www.linkedin.com/feed/hashtag/?keywords=systemdesign&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A6939603653020303360) [#coding](https://www.linkedin.com/feed/hashtag/?keywords=coding&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A6939603653020303360) [#interviewtips](https://www.linkedin.com/feed/hashtag/?keywords=interviewtips&highlightedUpdateUrns=urn%3Ali%3Aactivity%3A6939603653020303360)  
.