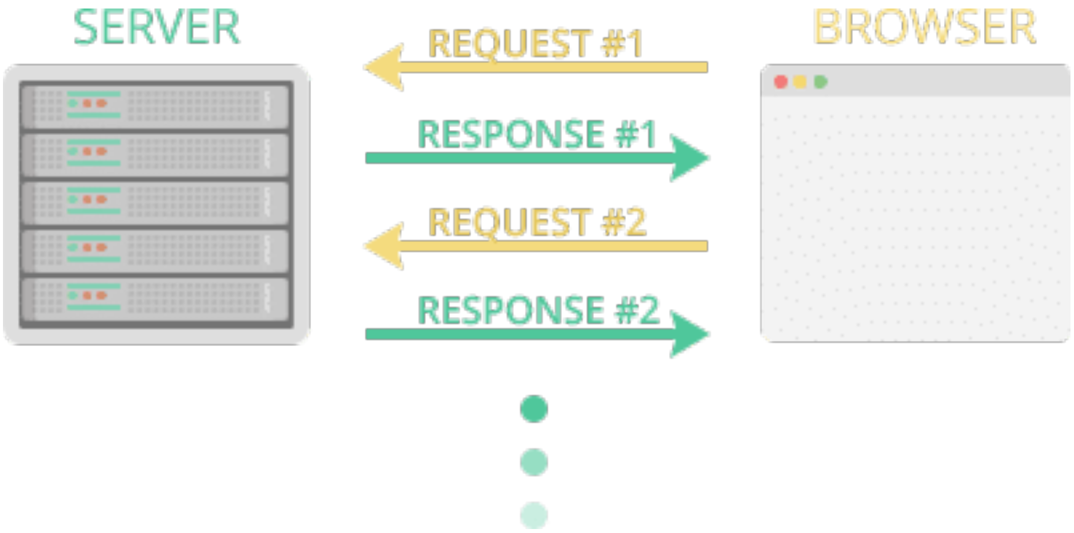
NGINX: Enable HTTP2 Protocol

NGINX : *Web-Server & Load Balancer*

- The first documented version of HTTP was released in 1991 as HTTP0.9, which later led to the official introduction and recognition of HTTP1.0 in 1996. HTTP1.1 followed in 1997 and has since received little iterative improvements.
- In February 2015, the Internet Engineering Task Force (IETF) HTTP Working Group revised HTTP and developed the second major version of the application protocol in the form of HTTP/2.

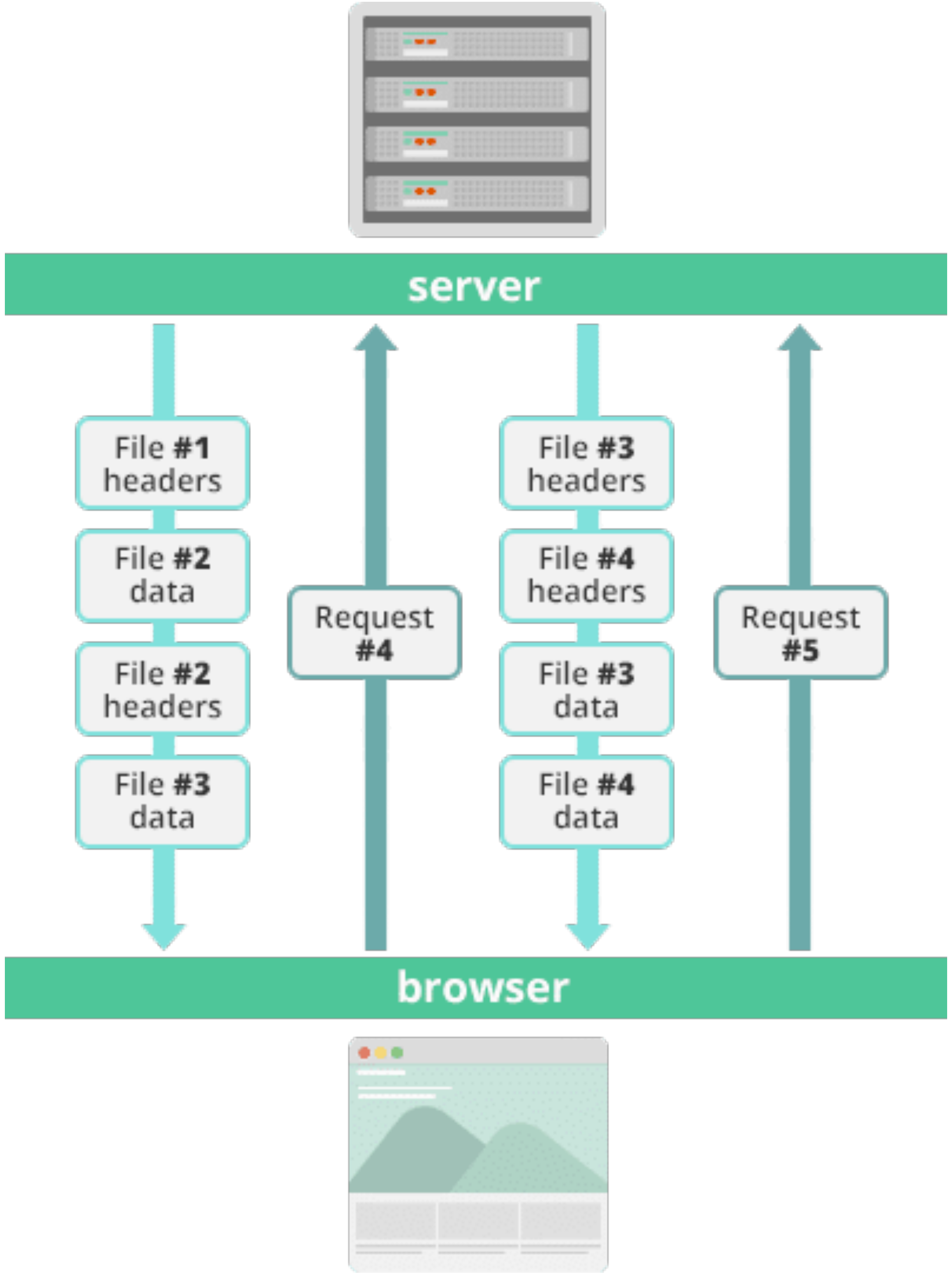
- **Why HTTP2 -**
- HTTP1.1 was limited to processing only one outstanding request per TCP connection, forcing browsers to use multiple TCP connections to process multiple requests simultaneously.
- Web browsers using multiple connections to process additional requests occupy a greater share of the available network resources, hence downgrading network performance for other users.



NGINX : Web-Server & Load Balancer

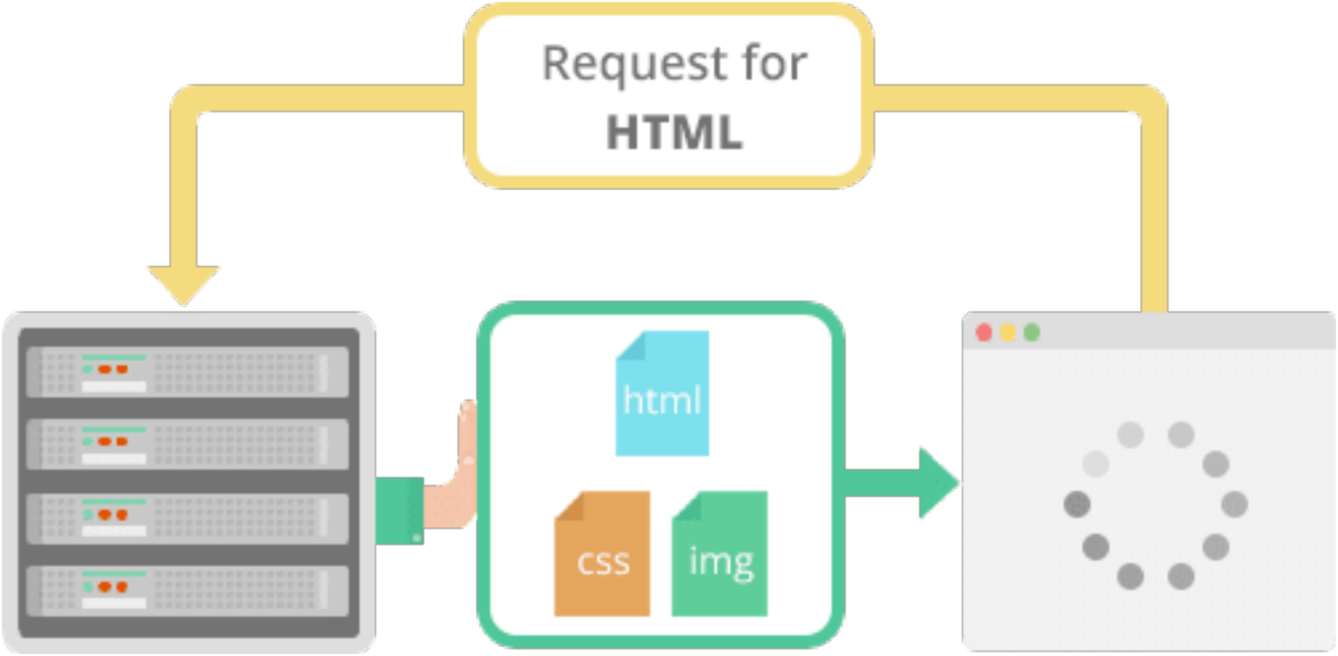
- **HTTP/2 Feature Upgrades -**
- **Multiplexed streams -** Bi-directional sequence of text format frames sent over the HTTP/2 protocol exchanged between the server and client are known as “streams”.
- Earlier iterations of the HTTP protocol were capable of transmitting only one stream at a time along with some time delay between each stream transmission.
- HTTP2 helped server to disintegrate the HTTP payload into small, independent and manageable interleaved sequence of frames. This information is then reassembled at the other end.
- With this capability, data packages from multiple streams are essentially mixed and transmitted over a single TCP connection. These packages are then split at the receiving end and presented as individual data streams. Transmitting multiple parallel requests simultaneously using HTTP version 1.1 or earlier required multiple TCP connections

NGINX : Web-Server & Load Balancer



NGINX : Web-Server & Load Balancer

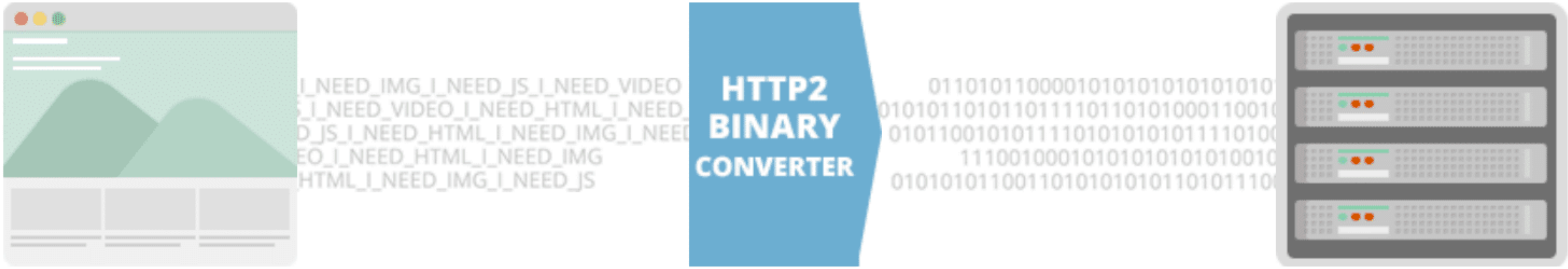
- **HTTP/2 Feature Upgrades -**
- **HTTP/2 Server Push -** This capability allows the server to send additional cacheable information to the client that isn't requested but is anticipated in future requests.



- This mechanism saves a request-respond round trip and reduces network latency.
- The server can prioritize pushed resources – a key performance differentiator in HTTP/2 vs HTTP1.

NGINX : Web-Server & Load Balancer

- **HTTP/2 Feature Upgrades -**
- **Binary Protocols -** HTTP1.x used to process text commands to complete request-response cycles. HTTP/2 will use binary commands to execute the same tasks.


I_NEED_IMG_I_NEED_JS_I_NEED_VIDEO
S_I_NEED_VIDEO_I_NEED_HTML_I_NEED_
D_JS_I_NEED_HTML_I_NEED_IMG_I_NEED
EO_I_NEED_HTML_I_NEED_IMG
HTML_I_NEED_IMG_I_NEED_JS
HTTP2
BINARY
CONVERTER

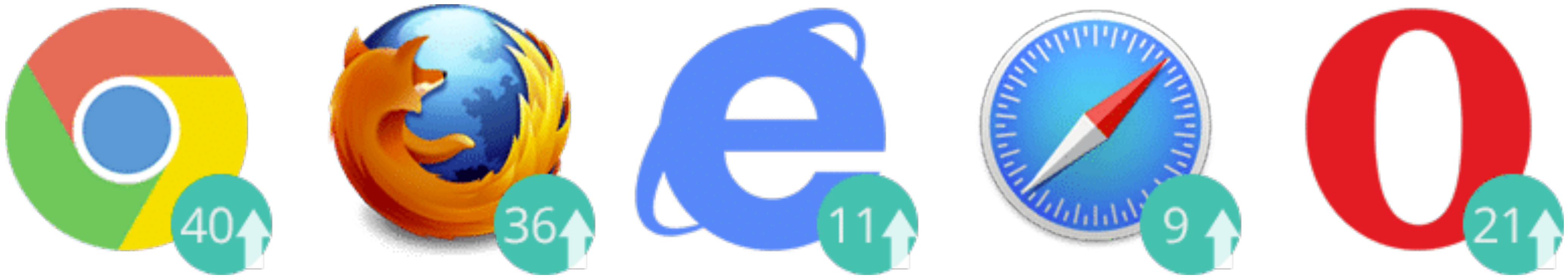
- Effective network resource utilization.
- Eliminating security concerns associated with the textual nature of HTTP1.x such as response splitting attacks.
- Efficient and robust in terms of processing of data between client and server.

NGINX : *Web-Server & Load Balancer*

- **HTTP/2 Feature Upgrades -**
- **Stateful Header Compression -** It uses the HPACK specification as a simple and secure approach to header compression. Both client and server maintain a list of headers used in previous client-server requests.
- HPACK compresses the individual value of each header before it is transferred to the server, which then looks up the encoded information in list of previously transferred header values to reconstruct the full header information.

NGINX : Web-Server & Load Balancer

- **Benefits of HTTP/2 -**
  - Web Performance
  - Mobile Web Performance
  - Cheaper Internet
  - Expansive Reach
  - Media Rich Experience
  - Improved Technology Utilization
  - Security
- **HTT2 Client Support -**



Will see you in Next Lecture…

See you in next lecture …