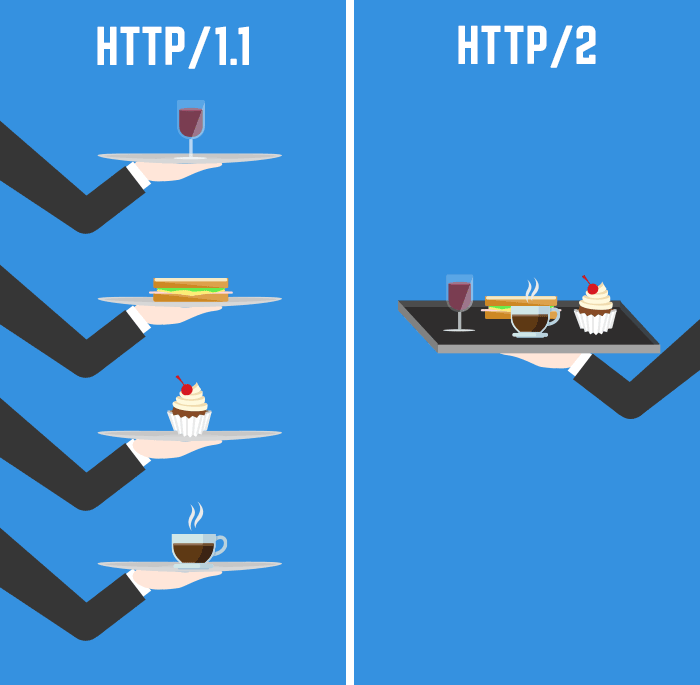
**1.HTTP1.1 vs HTTP2**

**HTTP 1.1:**

1. Introduced in 1997
2. Developed by Timothy Berners-Lee in 1989 as a communication standard for the World Wide Web
3. The first response that a client receives on an HTTP GET request is often not the fully rendered page. Instead, it contains links to additional resources needed by the requested page.
4. In HTTP/1.1, flow control relies on the underlying TCP connection.
5. In HTTP/1.1, if the developer knows in advance which additional resources the client machine will need to render the page, they can use a technique called resource inlining to include the required resource directly within the HTML document that the server sends in response to the initial GET request.
6. Programs like [gzip](https://www.gzip.org/) have long been used to compress the data sent in HTTP messages, especially to decrease the size of CSS and JavaScript files.

**HTTP 2:**

1. Introduced in 2015
2. HTTP/2 began as the SPDY protocol, developed primarily at Google with the intention of reducing web page load latency by using techniques such as compression, multiplexing, and prioritization.
3. In HTTP/2, the binary framing layer encodes requests/responses and cuts them up into smaller packets of information, greatly increasing the flexibility of data transfer.
4. HTTP/2 multiplexes streams of data within a single TCP connection.
5. Since HTTP/2 enables multiple concurrent responses to a client’s initial GET request, a server can send a resource to a client along with the requested HTML page, providing the resource before the client asks for it. This process is called server push.
6. One of the themes that has come up again and again in HTTP/2 is its ability to use the binary framing layer to exhibit greater control over finer detail.
7. <https://http2.github.io/> This is home page of HTTP/2, a major revision of the Web’s protocol.



**2. Objects And Its Internal Representation In JavaScript:**

A keyboard is an object, with properties. A keyboard has a color, a design, weight, a material it is made of, etc. The same way, JavaScript objects can have properties, which define their characteristics.

The syntax for an object using an object initializer is:

var obj = {

property\_1: value\_1, // property name may be an identifier...

2: value\_2, // or a number...

// ...,

'property n': value\_n // or a string

};

You access the properties of an object with a simple dot-notation:

objectName.propertyName

**Example of Initializing an object:**

var myPhone = {

make: 'Orange',

model: '2022',

color: ‘Blue’

};

We access the properties of an object with a simple dot-notation:

objectName.propertyName