HTTP/1.1 相較於 HTTP/1.0 協議的區別主要體現在：

1 快取處理

2 頻寬優化及網路連線的使用

3 錯誤通知的管理

4 訊息在網路中的傳送

5 網際網路地址的維護

6 安全性及完整性

常用的請求方式

GET 請求獲取 Request-URI 所標識的資源

POST 在 Request-URI 所標識的資源後附加新的資料

HEAD 請求獲取由 Request-URI 所標識的資源的響應訊息報頭

PUT 請求伺服器儲存一個資源，並用 Request-URI 作為其標識

DELETE 請求伺服器刪除 Request-URI 所標識的資源

TRACE 請求伺服器回送收到的請求資訊，主要用於測試或診斷

CONNECT 保留將來使用

OPTIONS 請求查詢伺服器的效能，或者查詢與資源相關的選項和需求

GET 方法：在瀏覽器的位址列中輸入網址的方式訪問網頁時，瀏覽器採用 GET 方法向伺服器獲取資源，POST 方法要求被請求伺服器接受附在請求後面的資料，常用於提交表單。GET 是用於獲取資料的，POST 一般用於將資料發給伺服器之用。

下面總結了 HTTP2.0 協議的幾個特性。

多路複用 (Multiplexing)

多路複用允許同時通過單一的 HTTP/2 連線發起多重的請求-響應訊息。在 HTTP/1.1 協議中瀏覽器客戶端在同一時間，針對同一域名下的請求有一定數量限制。超過限制數目的請求會被阻塞。這也是為何一些站點會有多個靜態資源 CDN 域名的原因之一，拿 Twitter 為例，http://twimg.com，目的就是變相的解決瀏覽器針對同一域名的請求限制阻塞問題。而 HTTP/2 的多路複用 (Multiplexing) 則允許同時通過單一的 HTTP/2 連線發起多重的請求-響應訊息。因此 HTTP/2 可以很容易的去實現多流並行而不用依賴建立多個 TCP 連線，HTTP/2 把 HTTP 協議通訊的基本單位縮小為一個一個的幀，這些幀對應著邏輯流中的訊息。並行地在同一個 TCP 連線上雙向交換訊息。

二進位制分幀

HTTP/2 在 應用層(HTTP/2)和傳輸層(TCP or UDP)之間增加一個二進位制分幀層。在不改動 HTTP/1.x 的語義、方法、狀態碼、URI 以及首部欄位的情況下, 解決了 HTTP1.1 的效能限制，改進傳輸效能，實現低延遲和高吞吐量。在二進位制分幀層中， HTTP/2 會將所有傳輸的資訊分割為更小的訊息和幀（frame）,並對它們採用二進位制格式的編碼 ，其中 HTTP1.x 的首部資訊會被封裝到 HEADER frame，而相應的 Request Body 則封裝到 DATA frame 裡面。

HTTP/2 通訊都在一個連線上完成，這個連線可以承載任意數量的雙向資料流。在過去， HTTP 效能優化的關鍵並不在於高頻寬，而是低延遲。TCP 連線會隨著時間進行自我調諧，起初會限制連線的最大速度，如果資料成功傳輸，會隨著時間的推移提高傳輸的速度。這種調諧則被稱為 TCP 慢啟動。由於這種原因，讓原本就具有突發性和短時性的 HTTP 連線變的十分低效。HTTP/2 通過讓所有資料流共用同一個連線，可以更有效地使用 TCP 連線，讓高頻寬也能真正的服務於 HTTP 的效能提升。

這種單連線多資源的方式，減少服務端的連結壓力,記憶體佔用更少,連線吞吐量更大；而且由於 TCP 連線的減少而使網路擁塞狀況得以改善，同時慢啟動時間的減少,使擁塞和丟包恢復速度更快。

首部壓縮（Header Compression）

HTTP/1.1 並不支援 HTTP 首部壓縮，為此 SPDY 和 HTTP/2 應運而生， SPDY 使用的是通用的 DEFLATE 演算法，而 HTTP/2 則使用了專門為首部壓縮而設計的 HPACK 演算法。

服務端推送（Server Push）

服務端推送是一種在客戶端請求之前傳送資料的機制。在 HTTP/2 中，伺服器可以對客戶端的一個請求傳送多個響應。Server Push 讓 HTTP1.x 時代使用內嵌資源的優化手段變得沒有意義；如果一個請求是由你的主頁發起的，伺服器很可能會響應主頁內容、logo 以及樣式表，因為它知道客戶端會用到這些東西。這相當於在一個 HTML 文件內集合了所有的資源，不過與之相比，伺服器推送還有一個很大的優勢：可以快取！也讓在遵循同源的情況下，不同頁面之間可以共享快取資源成為可能。