تعمل خوادم الويب على تمكين وصول
HTTP
إلى "موقع ويب" ، وهو عبارة عن مجموعة من المستندات والمعلومات الأخرى المنظمة في بنية شجرية ، مثل نظام ملفات الكمبيوتر. بالإضافة إلى توفير الوصول إلى المستندات الثابتة ، تقوم خوادم الويب الحديثة بتنفيذ مجموعة متنوعة من البروتوكولات لتمرير الطلبات إلى تطبيقات البرامج المخصصة التي توفر الوصول إلى المحتوى الديناميكي.

Lecture 03- Web Servers

Web servers enable HTTP access to a 'Web site,' which is simply a collection of documents and other information organized into a tree structure, much like a computer's file system. In addition to providing access to static documents, modern Web servers implement a variety of protocols for passing requests to custom software applications that provide access to dynamic content.

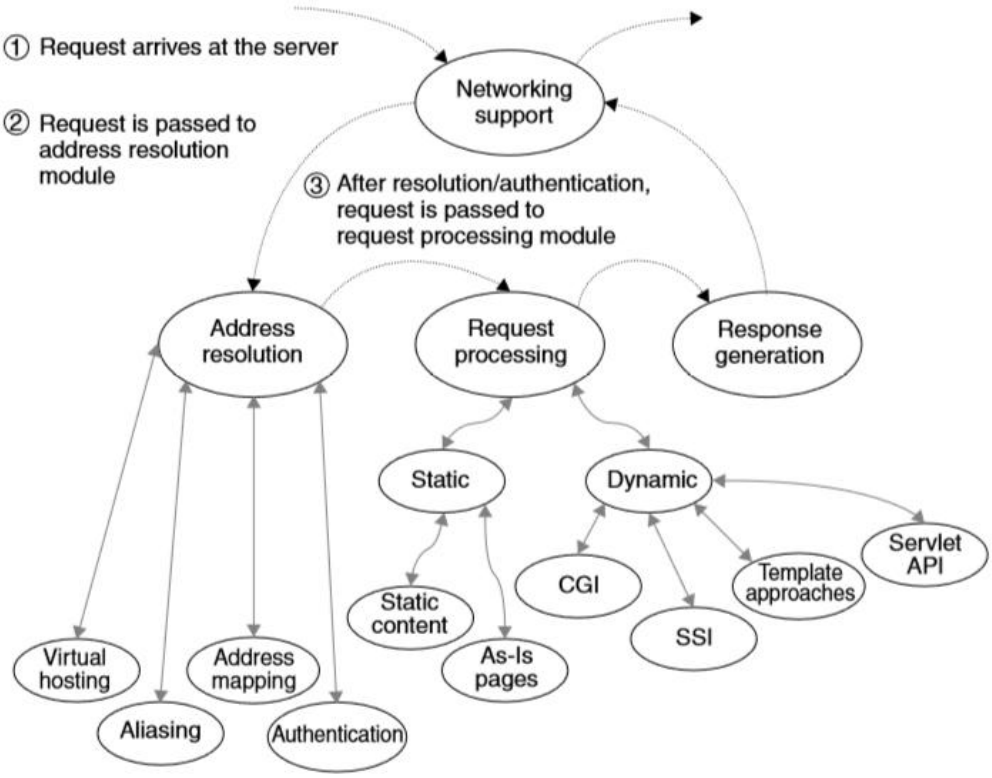
BASIC OPERATION

تتواصل خوادم الويب والمتصفحات والوكلاء عن طريق تبادل رسائل HTTP
يقوم الخادم بتنفيذ المهام التالية :

Web servers, browsers, and proxies communicate by exchanging HTTP messages. The server is performing the following tasks:

1. Receives and interprets HTTP requests, يتلقى ويفسر طلبات HTTP ،
2. Locates and accesses requested resources, يحدد ويصل إلى الموارد المطلوبة
3. and generates responses, which it sends back to the originators of the requests. يولد ردودًا ، والتي ترسلها مرة أخرى إلى منشئي الطلبات.

في الرسم البياني التالي ، يمكننا أن نرى كيف يعالج خادم الويب الطلبات الواردة ، ويولد ردودًا صادرة ، ويرسل تلك الردود مرة أخرى إلى مقدمي الطلبات المناسبين.
In the following diagram we can see how a Web server processes incoming requests, generates outgoing responses, and transmits those responses back to the appropriate requestors.



The Networking module is responsible for both receiving requests and transmiZtting responses over the network. When it receives a request, it must first pass it to the Address Resolution module, which is responsible for analyzing and 'pre-processing' the request. This pre-processing includes:

" وحدة الشبكات مسؤولة عن تلقي الطلبات وإرسال الاستجابات عبر الشبكة. عندما يتلقى طلبًا ، يجب عليه أولاً تمريره إلى وحدة تحليل العنوان ، المسؤولة عن تحليل الطلب و معالجته مسبقًا". تشمل هذه المعالجة المسبقة:

إذا كان خادم الويب هذا يوفر خدمة لعدة مجالات ، فحدد المجال الذي يستهدف هذا الطلب ، واستخدم المجال المكتشف لتحديد معلمات التكوين استضافة افتراضية

1. Virtual Hosting: if this Web server is providing service for multiple domains, determine the domain for which this request is targeted, and use the detected domain to select configuration parameters.

تحديد ما إذا كان هذا طلبًا لمحتوى ثابت أو ديناميكي ، استنادًا إلى مسار URL

تعيين العنوان

ومعلمات تكوين الخادم المحددة ، وحل العنوان إلى موقع فعلي داخل نظام ملفات الخادم

2. Address Mapping: determine whether this is a request for static or dynamic content, based on the URL path and selected server configuration parameters, and resolve the address into an actual location within the server's file system.

إذا كان المورد المطلوب محميًا ، فافحص بيانات اعتماد التفويض لمعرفة ما إذا كان الطلب واردًا من مستخدم مرخص

3. Authentication: if the requested resource is protected, examine authorization credentials to see if the request is coming from an authorized user.

بمجرد اكتمال المعالجة المسبقة ، يتم تمرير الطلب إلى وحدة معالجة الطلبات ، والتي تستدعي الوحدات الفرعية لتقديم محتوى ثابت أو ديناميكي حسب الاقتضاء. عندما تكمل الوحدة الفرعية المحددة معالجتها ، فإنها تمرر النتائج إلى وحدة توليد الاستجابة ، والتي تبني الاستجابة وتوجهها إلى وحدة الشبكات للإرسال.

Once the pre-processing is complete, the request is passed to the Request Processing module, which invokes sub-modules to serve static or dynamic content as appropriate. When the selected sub-module completes its processing, it passes results to the Response Generation module, which builds the response and directs it to the Networking module for transmission.

HTTP request processing

دعونا نعود خطوة إلى الوراء ونتذكر ما يجب أن يحدث لطلب HTTP للوصول إلى الخادم.

Let us take a step back and recollect what has to happen for an HTTP request to arrive at the server. For the purposes of this example, we shall examine a series of transactions in which an end-user is visiting her friend's personal web site found at http://mysite.org/. The process begins when the end user tells the browser to access the page found at the URL http://mysite.org/pages/simple-page.html.

لأغراض هذا المثال ، سنقوم بفحص سلسلة من المعاملات التي يزور فيها المستخدم النهائي موقع الويب الشخصي لصديقته الموجود على http://mysite.org/.

تبدأ العملية عندما يخبر المستخدم النهائي المتصفح بالوصول إلى الصفحة الموجودة في عنوان URL http://mysite.org pages / simple page.html.

GET http://mysite.org/pages/simple-page.html HTTP/1.1

Host: mysite.org

User-Agent: Mozilla/4.75 [en] (WinNT; U)

When the browser successfully receives and renders the page, the user sees that it has links to two other pages, which contain her friend's 'school links' (school.html) and 'home links' (home.html). Suppose now that the end user follows the link to her friend's 'school links' page

عندما يتلقى المتصفح الصفحة ويعرضها بنجاح ، يرى المستخدم أنه يحتوي على روابط لصفحتين أخريين ، والتي تحتوي على "روابط المدرسة" الخاصة بصديقتها (school.html) و "الروابط الرئيسية" (home.html).

لنفترض الآن أن المستخدم النهائي يتابع الرابط المؤدي إلى صفحة "روابط المدرسة" الخاصة بصديقتها

```
<HTML>
<HEAD><TITLE>Simple Page</TITLE></HEAD>
    <BODY>
        <H2>My Links</H2>
        <ul>
        <li><a href ="school.html">My school links</a></li>
        <li><a href ="home.html">My home links</a></li>
        </ul>
    </BODY>
</HTML>
```

Notice that the request does not contain the Connection: close header. This means that, if possible, the connection to the server should be left open so that it may be used to transmit further requests and receive responses. However, there is no guarantee that the connection will still be open at the time the user requests school.html.

لاحظ أن الطلب لا يحتوي على رأس الاتصال: إغلاق. وهذا يعني أنه إذا أمكن ، يجب ترك الاتصال بالخادم مفتوحًا بحيث يمكن استخدامه لإرسال المزيد من الطلبات وتلقي الردود. ومع ذلك ، ليس هناك ما يضمن أن الاتصال سيظل مفتوحًا في الوقت الذي يطلب فيه المستخدم school.html.

بحلول ذلك الوقت ، قد يكون الخادم أو الوكيل أو حتى المتصفح نفسه قد كسره. صُممت الاتصالات المستمرة لتحسين الأداء ، ولكن لا ينبغي أبدًا الاعتماد عليها في منطق التطبيق.

By that time, the server, a proxy, or even the browser itself might have broken it. Persistent connections are designed to improve performance, but should never be relied upon in application logic.

إذا كان الاتصال لا يزال مفتوحًا ، فسيستخدمه المتصفح لإرسال الطلب لصفحة ارتباطات المدرسة

If the connection is still open, the browser uses it to submit the request for the school links page

GET http://mysite.org/pages/school.html HTTP/1.1

Host: mysite.org

User-Agent: Mozilla/4.75 [en] (WinNT; U)

خلاف ذلك ، يجب أن يقوم المستعرض أولاً بإعادة الاتصال. اعتمادًا على تكوين المتصفح ، قد يحاول إما إنشاء ملفالاتصال بالخادم أو الاتصال عبر وكيل. وبالتالي ، يتلقى الخادم الطلب إما مباشرة من المتصفح أو من الوكيل

Otherwise, the browser must first re-establish the connection. Depending on the browser configuration, it may either attempt to establish a direct connection to the server or connect via a proxy. Consequently, the server receives the request either directly from the browser or from the proxy.

للاتصالات المستمرة ، يكون الخادم مسؤولاً عن الاحتفاظ بقوائم انتظار الطلبات والاستجابات. يحدد HTTP / 1.1 أنه في سياق اتصال واحد مفتوح باستمرار ، يمكن إرسال سلسلة من الطلبات. كما تحدد أنه يجب إرسال الردود على هذه الطلبات مرة أخرى بترتيب وصول الطلب (FIFO).

For persistent connections, the server is responsible for maintaining queues of requests and responses. HTTP/1.1 specifies that within the context of a single continuously open connection, a series of requests may be transmitted. It also specifies that responses to these requests must be sent back in the order of request arrival (FIFO).

One common solution is for the server to maintain both the input and output queues of requests. When a request is submitted for processing, it is removed from the input queue and inserted on the output queue. Once the processing is complete, the request is marked for release, but it remains on the output queue while at least one of its predecessors is still there. When all of its predecessors have gone from the output queue it is released, and its associated response is sent back to the browser either directly or through a proxy. Once the request is picked from the queue, the server resolves the request URL to the physical file location and checks whether the requested resource requires authentication. If the authentication fails, the server aborts further processing and generates the response indicating an error condition. If the authentication is not necessary or is successful, the server decides on the kind of processing required.

تسليم صفحات المحتوى الثابت

Delivery of Static content pages

بالنسبة لصفحة المحتوى الثابتة ، يقوم الخادم بتعيين عنوان URL إلى موقع ملف متعلق بجذر مستند الخادم. في المثال المقدم سابقًا ، قمنا بزيارة صفحة

For a static content page, the server maps the URL to a file location relative to the server document root. In the example presented earlier ,we visited a page

أحد الحلول الشائعة هو أن يحتفظ الخادم بقوائم انتظار الإدخال والإخراج للطلبات. عند تقديم طلب للمعالجة ، تتم إزالته من قائمة انتظار الإدخال وإدراجه في قائمة انتظار الإخراج. بمجرد اكتمال المعالجة ، يتم وضع علامة على الطلب للإفراج عنه لكنه يظل في قائمة انتظار الإخراج بينما لا يزال أحد أسلافه على الأقل موجودًا. عندما يذهب جميع أسلافه من قائمة ، انتظار الإخراج ، يتم تحريره ، ويتم إرسال الاستجابة المرتبطة به مرة أخرى إلى المتصفح إما مباشرة أو من خلال وكيل. بمجرد اختيار الطلب من قائمة الانتظار ، يحل الخادم عنوان URL للطلب إلى موقع الملف الفعلي ويتحقق مما إذا كان المورد المطلوب يتطلب المصادقة. إذا فشلت المصادقة ، يقوم الخادم بإلغاء مزيد من المعالجة ويقوم بإنشاء استجابة تشير إلى حالة خطأ. إذا لم تكن المصادقة ضرورية أو كانت ناجحة ، يقرر الخادم نوع المعالجة المطلوبة.

```
HTTP/1.1 200 OK
Date: Tue, 29 May 2001 23:15:29 GMT
Last-Modified: Mon, 28 May 2001 15:11:01 GMT
Content-type: text/html
Content-length: 193
Server: Apache/1.2.5

<HTML>
    <HEAD>
    <TITLE>School Page</TITLE>
    </HEAD>
    <BODY>
        <H2>My Links</H2>
        <ul>
        <li><a href ="classes.html">My classes</a></li>
        <li><a href ="friends.html">My friends</a></li>
        </ul>
    </BODY>
```

على موقع الويب الشخصي لشخص ما ، موجود في
http://mysite.org
صفحات / مدرسة. لغة البرمجة. يتم تعيين جزء المسار من عنوان
URL\ ، هذا
/pages/school.h ml
إلى اسم ملف صريح داخل نظام الملفات المحلي للخادم. إذا تم تكوين خادم الويب بحيث يكون جذر المستند هو ،
/ www / doc ، فسيتم تعيين عنوان
URL
</HTML> هذا إلى ملف خادم على / www / doc / pages school.html.

on someone's personal web site, found at http://mysite.org/pages/school. html. The path portion of this URL, /pages/school.html, is mapped to an explicit file name within the server's local file system. If the Web server is configured so that the document root is /www/doc, then this URL will be mapped to a server file at /www/doc/pages/school.html.

For static pages, the server must retrieve the file, construct the response, and transmit it back to the browser. بالنسبة للصفحات الثابتة ، يجب على الخادم استرداد الملف وإنشاء الاستجابة وإرساله مرة أخرى إلى المتصفح

For persistent connections, the response first gets placed in the output queue before the transmission. The first line of the response contains the status code that summarizes the result of the operation. The server controls browser rendering of the response through the Content-Type header that is set to a MIME type. Setting MIME types for static files is controlled through server configuration. بالنسبة للاتصالات المستمرة ، يتم وضع الاستجابة أولاً في قائمة انتظار الإخراج قبل الإرسال. يحتوي السطر الأول من الاستجابة على رمز الحالة الذي يلخص نتيجة العملية. يتحكم الخادم في عرض المستعرض للاستجابة من خلال رأس
ContentType
MIME. الذي تم تعيينه على نوع

It is the server-side mapping that determines the Content-Type header of the response. It is this header, which determines how the browser should render the response content—not the file name suffix, or a browser heuristic based on content analysis. إن تعيين جانب الخادم هو الذي يحدد رأس نوع المحتوى للاستجابة. هذا العنوان هو الذي يحدد كيف يجب على المستعرض عرض الاستجابة - وليس لاحقة اسم الملف ، أو توجيه المستعرض بناءً على تحليل المحتوى.

With all this in mind, it is important that the server set the Content-Type header to the appropriate MIME type so that the browser can render the content properly. The server may also set the Content-Length header to instruct the browser about the length of the content. This header is optional, and may not be present for dynamic content, because it is difficult to determine the size of the response before its generation is complete. Still, if it is possible to predict the size of the response, it is a good idea to include the Content-Length header.

تسليم المحتويات الديناميكية
Delivery of Dynamic Contents

مع وضع كل هذا في الاعتبار ، من المهم أن يقوم الخادم بتعيين رأس نوع المحتوى على نوع
MIME
المناسب حتى يتمكن المستعرض من عرض المحتوى بشكل صحيح. قد يقوم الخادم أيضًا بتعيين رأس طول المحتوى لإرشاد المستعرض حول طول المحتوى. هذا العنوان اختياري ، وقد لا يكون موجودًا للمحتوى الديناميكي ، لأنه من الصعب تحديد حجم الاستجابة قبل اكتمال إنشائها. ومع ذلك ، إذا كان من الممكن التنبؤ بحجم الاستجابة ، فمن الجيد تضمين عنوان
Content-Length.

For dynamic content, the server must take an explicit programmatic action to generate a response, such as: بالنسبة للمحتوى الديناميكي ، يجب أن يتخذ الخادم إجراءً برمجيًا صريحًا لإنشاء استجابة ، مثل:

تنفيذ برنامج تطبيقي
1. The execution of an application program,

تضمين معلومات من ملف ثانوي ، أو
2. The inclusion of information from a secondary file, or

تفسير القالب.
3. The interpretation of a template.

هذا النمط من المعالجة يشمل
This mode of processing includes:

(CGI) برامج واجهة البوابة العامة
1. Common Gateway Interface (CGI) programs,

2. Server-Side Include (SSI) pages, تضمين جانب الخادم (SSI) ، صفحات

صفحات خادم جافا (JSP) ،
3. Java Server Pages (JSP),

صفحات الخادم النشطة (ASP)
4. Active Server Pages (ASP), and

تستخدم خوادم الويب مجموعة من لاحقات اسم الملف / ملحقاته وبادئات عنوان URL لتحديد آلية المعالجة التي يجب استخدامها لإنشاء استجابة
5. Java Servlets

Web servers use a combination of file name suffixes/extensions and URL prefixes to determine which processing mechanism should be used to generate a response.

By default, it is assumed that a URL should be processed as a request for a static content page. However, this is only one of a number of possibilities. A URL path beginning with /servlet/ might indicate that the target is a Java servlet. A URL path beginning with /cgi-bin/ might indicate that the target is a CGI script. A URL where the target filename ends in .cgi might indicate this as

بشكل افتراضي ، من المفترض أن تتم معالجة عنوان URL كطلب لصفحة محتوى ثابتة. ومع ذلك ، هذا ليس سوى واحد من عدد من الاحتمالات. قد يشير مسار URL الذي يبدأ بـ / servlet / إلى أن الهدف هو Java servlet. قد يشير مسار URL الذي يبدأ بـ / cgi-bin / إلى أن الهدف هو نص CGI. عنوان URL حيث ينتهي اسم الملف الهدف
قد يشير cgi إلى هذا كـ

قد تشير عناوين
URL
حيث ينتهي اسم الملف المستهدف بـ
.php
أو
.
cfm
إلى وجوب استدعاء آلية معالجة القالب
(مثل PHP أو Cold Fusion)

well. URLs where the target filename ends in .php or .cfm might indicate that a template processing mechanism (e.g. PHP or Cold Fusion) should be invoked.

ADVANCED FEATURES

الاستضافة الافتراضية هي القدرة على تعيين أسماء خوادم ونطاقات متعددة إلى عنوان
IP
واحد. كان عدم وجود دعم لهذه الميزة في
HTTP / 1.0
مشكلة صارخة لمزودي خدمة الإنترنت
(ISP).
بعد كل شيء ، هناك حاجة عند تسجيل اسم مجال جديد وتريد أن يدعمه مزود خدمة الإنترنت. تمتلك خوادم
HTTP / 1.1
عددًا من المسؤوليات فيما يتعلق بالاستضافة الافتراضية:

Virtual Hosting

Virtual hosting is the ability to map multiple server and domain names to a single IP address. The lack of support for such feature in HTTP/1.0 was a glaring problem for Internet Service Providers (ISP). After all, it is needed when you register a new domain name and want your ISP to support it. HTTP/1.1 servers have a number of responsibilities with regard to virtual hosting:

استخدم المعلومات الموجودة في رأس المضيف المطلوب لتعريف المضيف الظاهري

1. Use information in the required Host header to identify the virtual host.

قم بإنشاء استجابات خطأ باستخدام رمز حالة الطلب السيئ 400 المناسب في حالة عدم وجود المضيف.

2. Generate error responses with the proper 400 Bad Request status code in the absence of Host.

دعم عناوين
URL
المطلقة في الطلبات ، على الرغم من عدم وجود متطلبات بأن الخادم المحدد في عنوان
URL
المطلق يطابق رأس المضيف.

3. Support absolute URLs in requests, even though there is no requirement that the server identified in the absolute URL matches the Host header.

دعم العزل والتكوين المستقل لأشجار المستندات والتطبيق من جانب الخادم بين المضيفين الظاهريين المختلفين المدعومين من نفس تثبيت الخادم

4. Support isolation and independent configuration of document trees and server side applications between different virtual hosts that are supported by the same server installation.

تحويلات مقسمة

Chunked transfers

هناك عدد من المناسبات التي قضيت فيها دقائق طويلة جالسًا أمام متصفحك في انتظار صفحة بطيئة بشكل خاص. قد يكون ذلك بسبب الاتصال البطيء أو قد يكون تطبيق الخادم بطيئًا. في كلتا الحالتين ، عليك الانتظار على الرغم من أن كل ما تحتاجه قد يكون إلقاء نظرة سريعة على الصفحة قبل الانتقال.

The chances are there were a number of occasions when you spent long minutes sitting in front of your browser waiting for a particularly slow page. It could be because of the slow connection or it could be that the server application is slow. Either way you have to wait even though all you need may be to take a quick look at the page before you move on.

HTTP/1.1 specification introduced the notion of transfer encoding as well as the first kind of transfer encoding—chunked—that is designed to enable processing of partially transmitted messages. According to the specification, the server is obligated to decode HTTP requests containing the *Content-Transfer-Encoding: chunked* header prior to passing it to server applications. A similar obligation is imposed on the browser. Server applications, of course, may produce chunked responses, which are particularly recommended for slow applications.

Caching Support

نركز مناقشتنا على التزامات الخادم لدعم التخزين المؤقت للمتصفح وكذلك عناصر التحكم في الخادم فيما يتعلق بسلوكيات التخزين المؤقت للمتصفح ، حيث كانت هناك مشاكل في تنفيذ استراتيجيات التخزين المؤقت الأكثر تقدمًا:

We concentrate our discussion on server obligations in support of browser caching as well as server controls with regard to browser caching behaviors, there were, problems with implementing more advanced caching strategies:

1. On-request verification of cache entries meant doubling the number of requests for modified pages using HEAD requests. As you remember from our earlier HTTP discussion, HEAD requests result in response messages with empty bodies. At best, such responses contained enough information to submit GET requests.

2. HTTP/1.0 servers, as a rule, did not include the Last-Modified header in response messages, making it much harder to check whether cache entries remained current. Verification had to be based on unreliable heuristics (e.g., changes in content length, etc.).

3. There was no strict requirement for HTTP/1.0 servers to include the Date header in their responses (even though most did) making it harder to properly record cache entries.

HTTP/1.1 requires servers to comply with the following requirements in support of caching policies:

يتطلب
HTTP / 1.1
من الخوادم الامتثال للمتطلبات التالية لدعم سياسات التخزين المؤقت:

يجب أن تجري خوادم
HTTP / 1.1
التحقق من إدخال ذاكرة التخزين المؤقت عند تلقي الطلبات التي تتضمن "إذا تم التعديل منذ" و "إذا لم يتم تعديلها منذ" تم تعيين الرؤوس على تاريخ بتنسيق
GMT
(على سبيل المثال ، التاريخ: الأحد ، 23 مارس 1997 ، 22:15:51 بتوقيت جرينتش).

1. HTTP/1.1 servers must perform cache entry verification when receiving requests that include If-Modified-Since and If-Unmodified-Since headers set to a date in the GMT format (e.g. Date: Sun, 23 Mar 1997, 22:15:51 GMT).

يجب أن تتعامل العوائم التواريخ غير السالمة والمستقبلية ومعاملة إنشاء نفس الاستجابة التي كانت ستنشئها في حالة عدم وجود هذه الرؤوس إذا تم استيفاء الشرط (تم تعديل المحتوى في حالة عنوان If Modified-Since أو لم يتم تعديله في حالة If
.(غير معدلة منذ العنوان). العوائم مسؤولة أيضًا عن إنشاء أكواد الحالة المناسبة للحالات الفاشلة (304 غير معدل و 412 فشل الشرط المسبق في المقابل

Servers must ignore invalid and future dates and attempt to generate the same response they would in the absence of these headers if the condition is satisfied (content was modified in the case of the If-Modified-Since header or not modified in the case of the If-Unmodified-Since header). Servers are also responsible for generating proper status codes for failed conditions (304 Unmodified and 412 Precondition Failed correspondingly).

من المستحسن أن يبذل مطور الخادم جهدًا لتضمين عنوان
Last Modified
في رسائل الاستجابة كلما أمكن ذلك. تستخدم المتصفحات هذه القيمة للمقارنة مع التواريخ المخزنة بالمدخلات ذاكرة التخزين المؤقت.

2. It is recommended that server implementers make an effort to include the Last-Modified header in response messages whenever possible. Browsers use this value to compare against dates stored with cache entries.

3. Unlike HTTP/1.0, HTTP/1.1 servers are required to include the Date header with every response, which makes it possible to avoid errors that may happen when browsers rely on their own clocks.

على عكس
HTTP / 1.0
يُطلب من خوادم ،
HTTP / 1.1
تضمين رأس التاريخ مع كل استجابة ، مما يجعل من الممكن تجنب الأخطاء التي قد تحدث عندما تعتمد المتصفحات على ساعاتها الخاصة