HTTP RESPONSES

When a web server receives a HTTP request, it interprets it, and tries to fetch the data requested, and return it. It may well fail, but what ever the result of attempting to fulfil the request, the server will formulate a HTTP response to communicate the outcome of the request to the client.

Similar to the request, a HTTP response has three parts, a status line, zero or more response header lines, and a final optional data segment, separated from the headers by a blank line.

Below is a truncated version of the HTTP response from Allison's web server to a request:

```
HTTP/1.1 200 OK
Date: Sat, 09 May 2015 15:52:42 GMT
Server: Apache
X-Pingback: http://www.podfeet.com/blog/xmlrpc.php
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Set-Cookie: PHPSESSID=eand2g7q77privgcpvi6m7i7g2; path=/
Vary: Accept-Encoding
Transfer-Encoding: chunked
```

```
Content-Type: text/html; charset=UTF-8


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <title>NosillaCast</title>
...
```

The first line of the response gives the HTTP version, and most importantly, the HTTP response code. This tells the client what kind of response it is receiving. You could receive a successful response, a response instructing the client to re-issue its request to a different URL (i.e. a redirect), a request for authentication (a username and password popup), or an error message.

After the HTTP response line comes a list of HTTP header lines, again, we won't go into them all, but I do want to draw your attention to a few important ones. Firstly, the **Server** header makes it possible to gather statistics on the web servers in use on the internet – notice that Allison's site is powered by an Apache web server. The single most important response header is **Content-Type**, which tells the client what type of data it will receive after the blank line, and optionally, how it's encoded.

In this case, the data section contains HTML markup encoded using UTF-8. Also notice that the server is requesting the client set a new cookie using the **Set-Cookie** header, and that the **Cache-Control** header is telling the client, in many different ways, that it absolutely positively should not cache a copy of this page. The actual HTML markup for Allison's home page is hundreds of lines long, I have only shown the first six lines.

It's important to note that rendering a single web page generally involves many HTTP requests, often to multiple servers. The first response will usually be the HTML markup for the web page in question, but that HTML will almost certainly contain links to other resources need to render the page, like style sheets, images, JavaScript files, etc.. As an example, rendering Allison's home page requires 107 HTTP requests! That's on the high side because Allison has a lot of videos embedded in her home page, and quite a few widgets embedded in her sidebars. However, on the modern web it's not unusual to need this many requests to render a single page.

HTTP Response Codes

There are many supported HTTP response codes (click here for a full list), and we're not going to go into them all, but I do want to explain the way they are grouped, and highlight some common ones you're likely to come across.

HTTP response codes are three-digit numbers starting with 1, 2, 3, 4, or 5. They are grouped into related groups by their first digit. All response codes starting with a 1 are so-called informational responses. These are rarely used. All response codes starting with a 2 are successful responses to requests. All response codes starting with a 3 are redirection responses. All responses starting with a 4 are client errors (in a very loose sense), and finally, all responses starting with a 5 are server errors.

Some common HTTP response codes:

200 - OK

This is the response code you always hope to get, it means your request was successful

301 - Moved Permanently

A permanent redirect, this redirect may be cached by clients

302 - Found

A temporary redirect, this redirect should not be cached by clients, it could change at any time

400 - Bad Request

The HTTP request sent to the server was not valid. You're unlikely to ever see this in a browser, but if you muck around constructing your own requests on the terminal you might well see it when you get something wrong!

401 - Not Authorised

Tells the client to request a username and password from the user

403 - Forbidden

The requested URL exists, but the client has been denied access, perhaps based on the user they have logged in as, the IP address they are accessing the site from, or the file-type of the URL they are attempting to access.

404 - Not Found

One of the most common errors you'll see – your request was valid, the server understood it, but it has no content to return to you at that URL.

500 - Internal Server Error

The web programmers's most hated error – it just means the server encountered and error while trying to fulfil your request.

502 - Bad Gateway

In the days of CDNs (Content Delivery Networks), these errors are becoming ever more common. It means that your browser has successfully contacted a front-end web server, probably at the CDN, but that the back-end server that actually contains the information you need is not responding to the front-end server. The front-end server is considered a gateway to the backend server, hence the name of the error.

503 - Service Unavailable

The server is temporarily too busy to deal with you – effectively a request to try again later.

504 - Gateway Timeout

This error is similar to a 502, and is also becoming ever more common with the rise of CDNs, it means the backend server is up, but is responding too slowly to the front-end server, and the front-end server is giving up.

MIME Types

HTTP uses the **Content-Type** header to specify the type of data being returned. The value of that header must be a so-called MIME Type, or *internet media type*. MIME Types have their origins in the common suite of email protocols, and were later adopted for use on the world wide web – after all, why re-invent the wheel!?

There are MIME types for just about everything, and they consist of two parts, a general type, and then a more specific identifier. E.g. all the text-based code files used on the web have MIME types starting with **text**, e.g.:

text/html

HTML markup

text/javascript

JavaScript code

text/css

CSS Style Sheet definitions

Some other common web MIME Types include:

image/jpeg

JPEG Photos

image/png

PNG graphics

audio/mpeg

MP3 audio

video/mp4

MPEG 4 video

Source: https://www.bartbusschots.ie/s/2015/05/09/taming-the-terminal-part-34-of-n-introducing-http/