School of Information Technologies

COMP5347: Web Application Development

Week 4 Tutorial: HTTP and Browser Performance

# Learning Objectives

* Understand HTTP Protocol

Understand browser behavior with respect to requesting resources and rendering process

*•*

## Question 1: Examine the HTTP protocol using TELNET/SSH

Start PUTTY and set up a SSH session to ucpu1.ug.cs.usyd.edu.au (alternatively, you may use ucpu3.ug.cs.usyd.edu.au) using your unikey and password.

1. In the SSH window type the following command:

telnet soit-usrweb-pro-1.ucc.usyd.edu.au 80

GET /~cshe6391/week4.html HTTP/1.0

You need to type Enter key an extra time after the GET command to send out the request. This will send out an HTTP 1.0 request to the server soit-usrweb-pro-1.ucc.usyd.edu.au Examine the response message and identify the following fields from the status lines:

|  |  |
| --- | --- |
| Protocol Version | HTTP1.1 |
| Status Code | 200 |
| Status Message | OK |

Also Identify the following fields from the header line:

|  |  |
| --- | --- |
| Web Server | Apache/2.2 15 (Red Hat) |
| ETag | 2c8003c-16c-5680ed05cab7d |
| Content Type | text/html; charset=UTF-8 |
| TCP connection closed? | close |

1. Now type the following command to send out an HTTP/1.1 request

telnet soit-usrweb-pro-1.ucc.usyd.edu.au 80

GET /~cshe6391/week4.html HTTP/1.1

Host: soit-usrweb-pro-1.ucc.usyd.edu.au

Remember to type Enter key twice to send out the request. You may have noticed that in the first request, the connection is closed immediately, while in the second one, the connection is still open and you are able to type in another request, that is, if your typing speed is really fast. The connection will be closed after some timeout period.

1. Now type the following command to send out a conditional request

GET /~cshe6391/week4.html HTTP/1.1

Host: soit-usrweb-pro-1.ucc.usyd.edu.au

If-None-Match: "2c8003c-16c-5680ed05cab7d"

Read the response message and identify the status code. You should expect an **“HTTP/1.1 304 Not Modified” status code and message. The response should not contain any body.**

## Question 2: Inspect browser HTTP request/response details with DevTool

1. Start Google Chrome and open the developer tool same as you did in week 3 lab. In this week, we focus on the Network tab instead. Type the following URL in the address bar:

http://soit-usrweb-pro-1.ucc.usyd.edu.au/~cshe6391/week4.html and type enter to send out the request. You should get a screen similar to the Figure 1

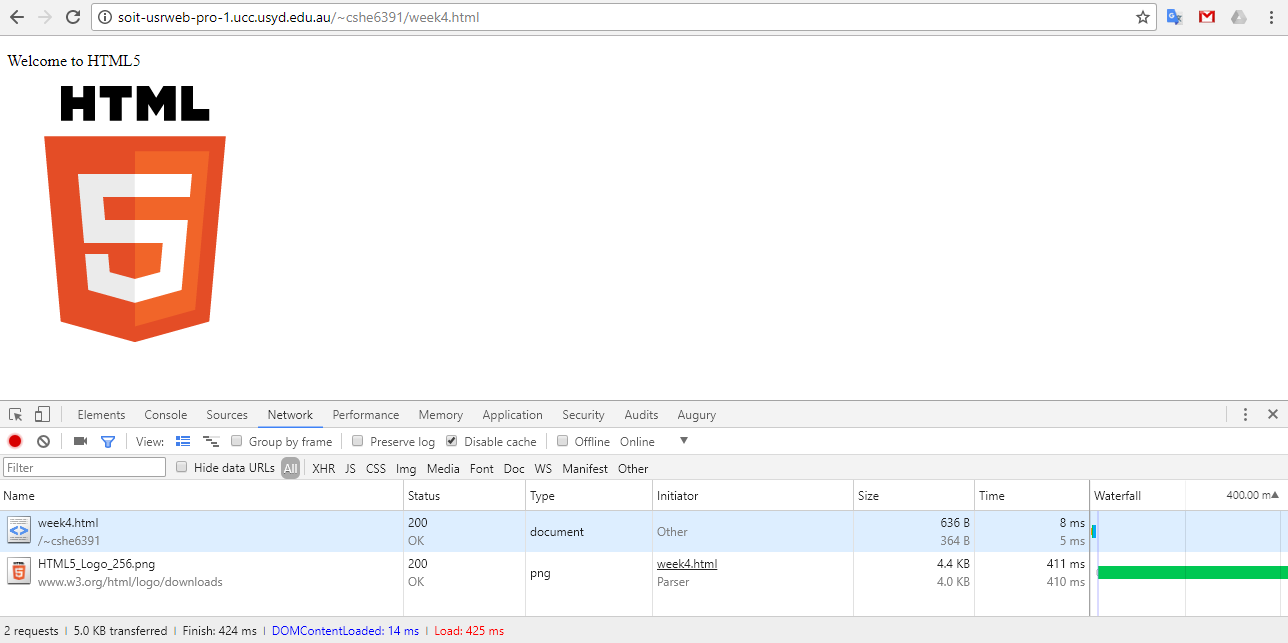


Figure 1: Screenshot of Network tab in DevTool

The network panel provides a number of views. Figure 1 does not show overview and use “small request rows” option. You may switch between different views by clicking the two option icons after the View: item.

You can inspect various details of each individual request by clicking the request and selecting what you want to view. For instance, Figure 2 shows the “timing” of request week4.html, Figure 3 shows the headers of request HTML5 Logo 256.png

Use the information provided by DevTool to answer the following question regarding HTTP messages:

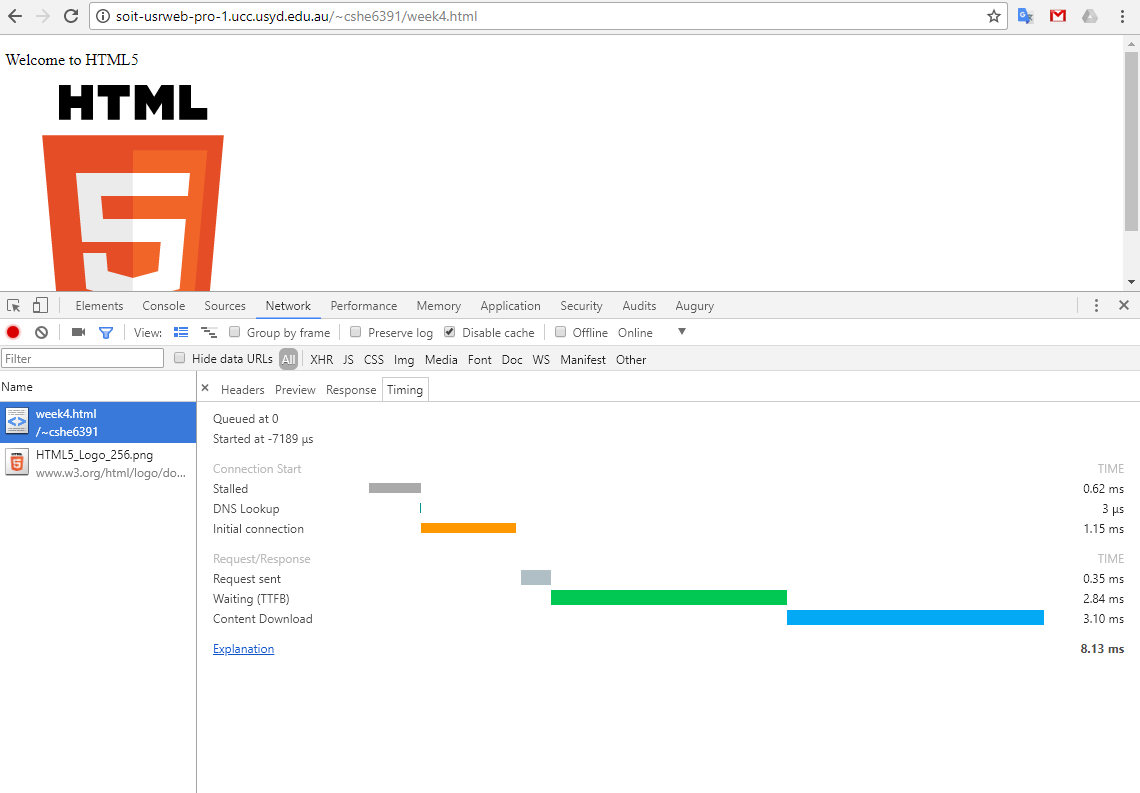


Figure 2: Timing details of request week4.html

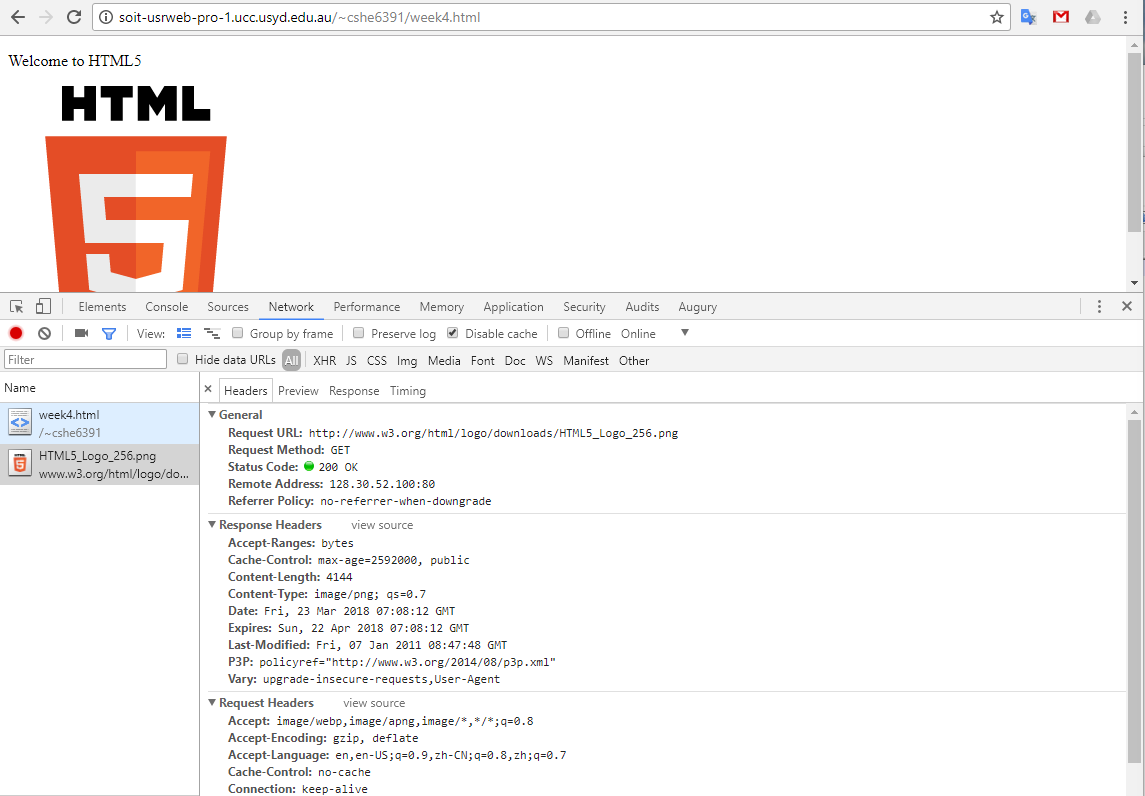


Figure 3: Headers of request HTML5 Logo 256.png

How many requests are sent by the browser? Are the requests sent to the same server?

*•*

*•*

What are the response status codes for all requests?

*•*

Which server sends cache instructions and what are the instructions?

*•*

Use the information provided by DevTool to answer the following questions regarding browser rendering process:

What are the DOMContentLoaded time and the Load time?

*•*

What happens between these two events?

*•*

In Figure 1, there is a small gap between week4.html request time (8ms) and the DOMContentLoaded event time (14ms). What happens during the gap?

*•*

In Figure 1, the request for HTML5 Logo 256.png took much 411ms while the request for week4.html took 8ms. You may have slightly different numbers in your DevTool. What is the main cause for delay in getting HTML5 Logo 256.png.

*•*

1. Open a new chrome tab and navigate to chrome://cache/. This will bring up all content cached locally by Chrome. Search for the two objects: week4.html and HTML5 Logo 256.png. You should be able to find both. Clicking the link representing each object will display the actual cache entry. You will find the data captured are similar to the respective HTTP response header. For instance, both entries contain basic data such as Last Modified time. The cache entry of HTML5 Logo 256.png contain the Cache-Control header and Expires time. The cache entry of week4.html contains ETag data.

Theoretically, the browser will use cache validation mechanism for week4.html and cash expiration mechanism for HTML5 Logo 256.png. We will see if that is the case.

1. Now reload the page and inspect the content of Network panel (which would be similar to Figure 4 to answer the following questions:

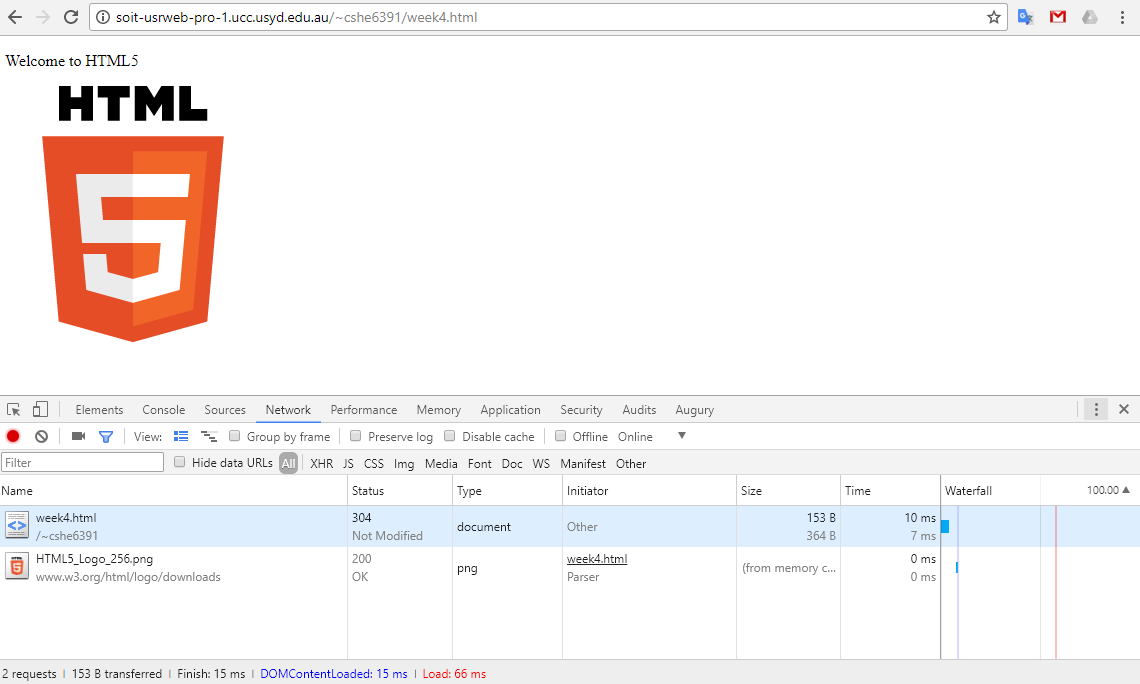


Figure 4: Reloading week4.html

what are the latencies for both requests? Are there any difference between the latencies observed in the previous question (first time requesting week4.html)

*•*

What are the DOMContentLoaded and load time? Are there any difference between the time recorded in the previous question (first time requesting week4.html)

*•*

How does the browser obtain content of week4.html and HTML5 Logo 256.png?

*•*

1. Open DevTool on another tab, point the browser to the following URL:

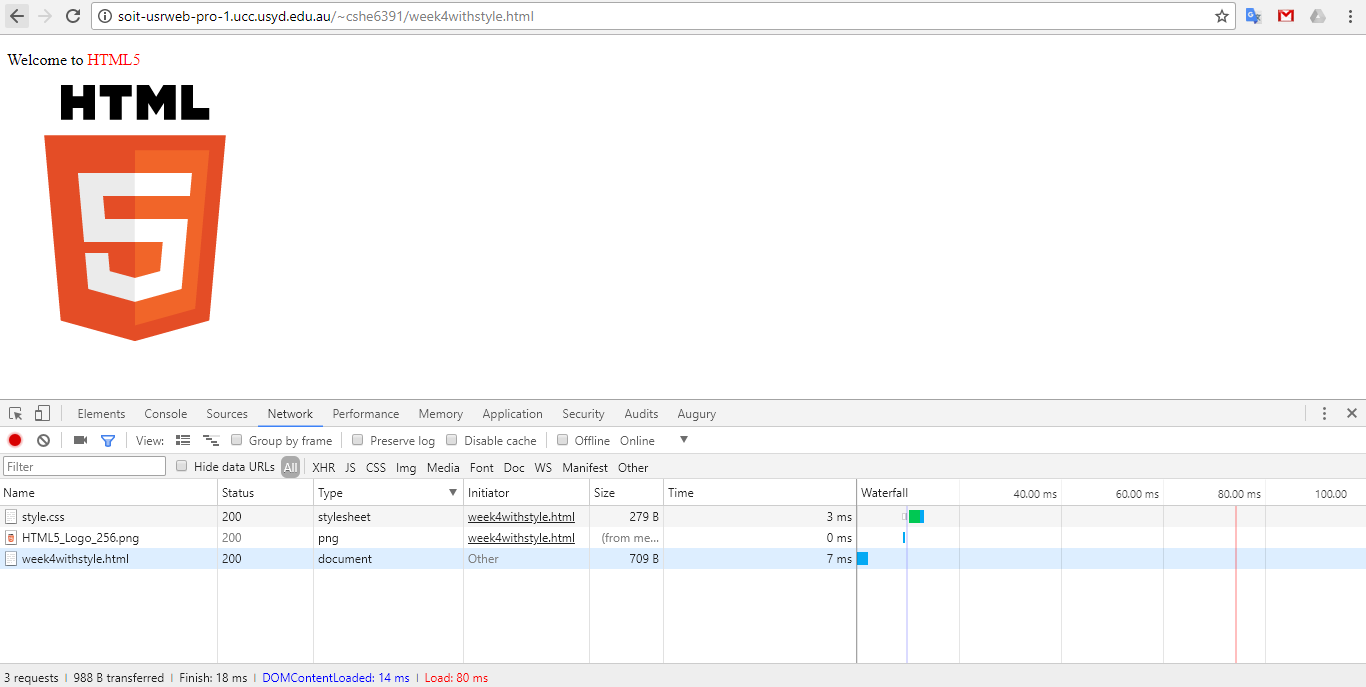
http://soit-usrweb-pro-1.ucc.usyd.edu.au/~cshe6391/week4withstyle.html Your network panel may look like Figure 5:

Figure 5: Loading week4 with style.html

Try to interpret the rendering process of the browser based on the DevTool information. This page has two supporting resources: style.css and HTML5 Logo 256.png. In which order does the browser send request for each?