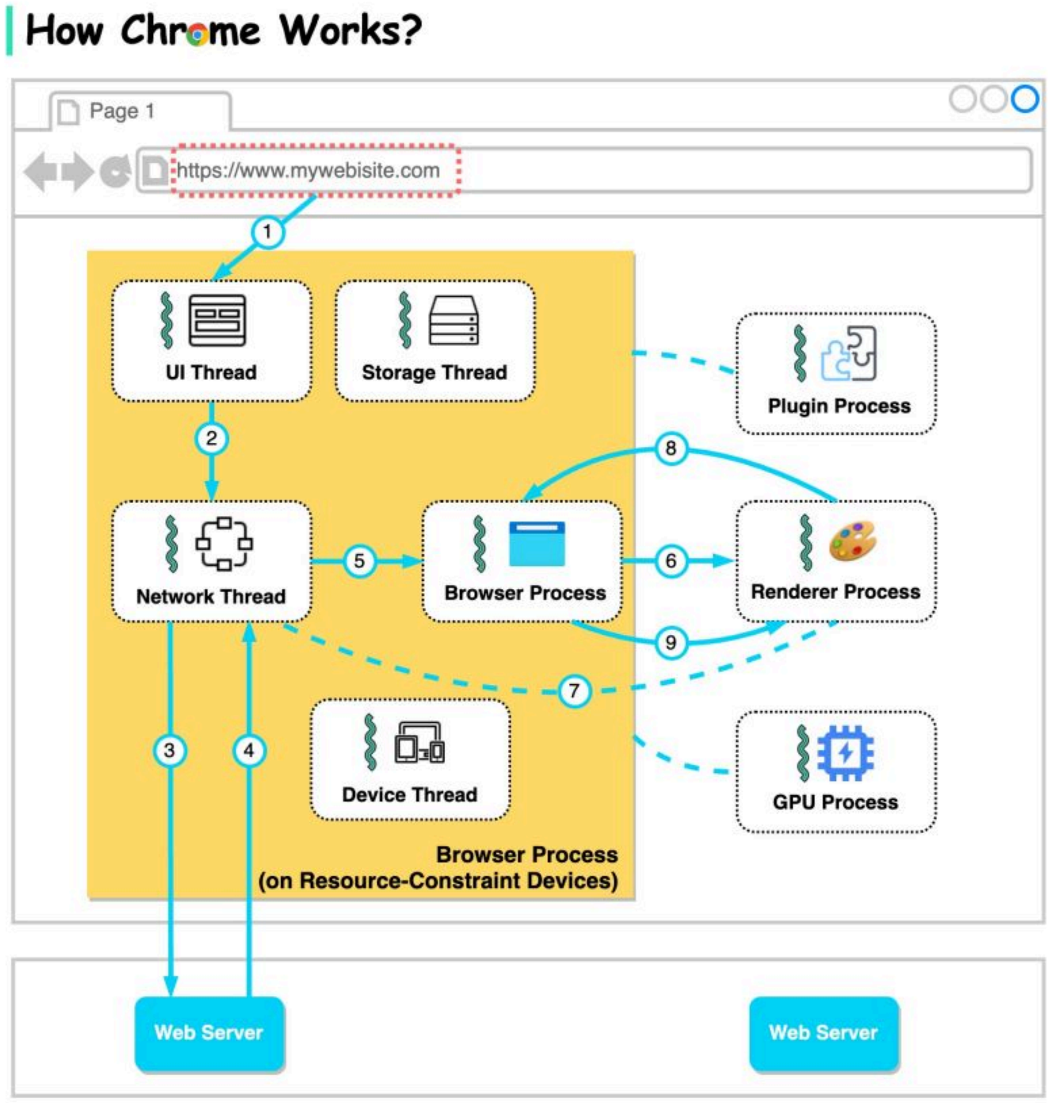
How does Chrome work?

The diagram below shows the architecture of a modern browser. It is based on our understanding of “Inside look at modern web browser” published by the chrome team.



Source: https://developer.chrome.com/blog/inside-browser-part1/

There are in general 4 processes: browser process, renderer process, GPU process, and plugin process.

- Browser process controls the address bar, bookmarks, back and forward buttons, etc.
- Renderer process controls anything inside of the tab where a website is displayed.
- GPU process handles GPU tasks.
- Plugin process controls the plugins used by the websites.

The browser process coordinates with other processes.

When Chrome runs on powerful hardware, it may split each service in the browser process into different threads, as the diagram below shows. This is called Servicification.

Now let’s go through the steps when we enter a URL in Chrome.

Step 1: The user enters a URL into the browser. This is handled by the UI thread.

Step 2: When the user hits enter, the UI thread initiates a network call to get the site content.

Steps 3-4: The network thread goes through appropriate network protocols and retrieves the content.

Step 5: When the network thread receives responses, it looks at the first few bytes of the stream. If it is an HTML file, it is passed to the renderer process by the browser process.

Steps 6-9: An IPC is sent from the browser process to the renderer process to commit the navigation. A data pipe is established between the network thread and the renderer process so that the renderer can receive data. Once the browser process hears confirmation that the commit has happened in the renderer process, the navigation is complete and the document loading phase begins.

Over to you: Why does Chrome assign each tab a renderer process?

Reference: Inside look at modern web browser