



# Chromium Blog

News and developments from the open source browser project

---

## Multi-process Architecture

Thursday, September 11, 2008

Unlike most current web browsers, Google Chrome uses many operating system processes to keep web sites separate from each other and from the rest of your computer. In this blog post, I'll explain why using a multi-process architecture can be a big win for browsers on today's web. I'll also talk about which parts of the browser belong in each process and in which situations Google Chrome creates new processes.

### 1. Why use multiple processes in a browser?

In the days when most current browsers were designed, web pages were simple and had little or no active code in them. It made sense for the browser to render all the pages you visited in the same process, to keep resource usage low.

Today, however, we've seen a major shift towards active web content, ranging from pages with lots of JavaScript and Flash to full-blown "web apps" like Gmail. Large parts of these apps run inside the browser, just like normal applications run on an operating system. Just like an operating system, the browser must keep these apps separate from each other.

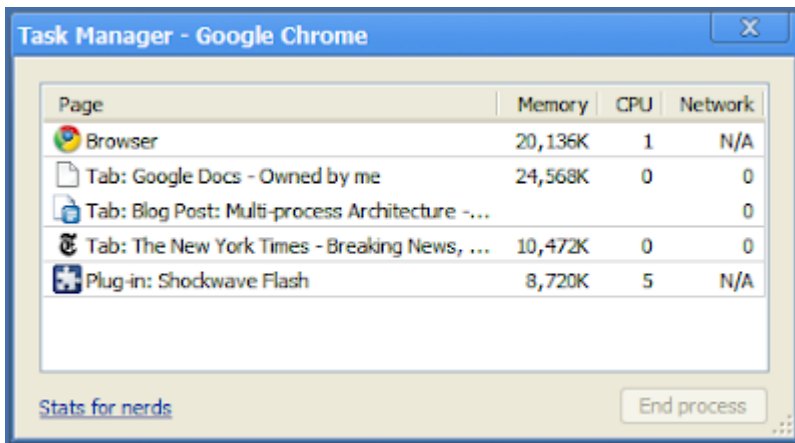
On top of this, the parts of the browser that render HTML, JavaScript, and CSS have become extraordinarily complex over time. These rendering engines frequently have bugs as they continue to evolve, and some of these bugs may cause the rendering engine to occasionally crash. Also, rendering engines routinely face untrusted and even malicious code from the web, which may try to exploit these bugs to install malware on your computer.

In this world, browsers that put everything in one process face real challenges for robustness, responsiveness, and security. If one web app causes a crash in the rendering engine, it will take the rest of the browser with it, including any other web apps that are open. Web apps often have to compete with each other for CPU time on a single thread, sometimes causing the entire browser to become unresponsive. Security is also a concern, because a web page that exploits a vulnerability in the rendering engine can often take over your entire computer.

It doesn't have to be this way, though. Web apps are designed to be run independently of each other in your browser, and they could be run in parallel. They don't need much access to your disk or devices, either. The security policy used throughout the web ensures this, so that you can visit most web pages without worrying about your data or your computer's safety. This means that it's possible to more completely isolate web apps from each other in the browser without breaking them. The same is true of browser plug-ins like Flash, which are loosely coupled with the browser and can be separated from it without much trouble.

Google Chrome takes advantage of these properties and puts web apps and plug-ins in separate processes from the browser itself. This means that a rendering engine crash in one web app won't affect the browser or other web apps. It means the OS can run web apps in parallel to increase their responsiveness, and it means the browser itself won't lock up if a particular web app or plug-in stops responding. It also means we can run the rendering engine processes in a restrictive sandbox that helps limit the damage if an exploit does occur.

Interestingly, using multiple processes means Google Chrome can have its own Task Manager (shown below), which you can get to by right clicking on the browser's title bar. This Task Manager lets you track resource usage for each web app and plug-in, rather than for the entire browser. It also lets you kill any web apps or plug-ins that have stopped responding, without having to restart the entire browser.



The screenshot shows the 'Task Manager - Google Chrome' window. It contains a table with the following data:

Page	Memory	CPU	Network
Browser	20,136K	1	N/A
Tab: Google Docs - Owned by me	24,568K	0	0
Tab: Blog Post: Multi-process Architecture - ...			0
Tab: The New York Times - Breaking News, ...	10,472K	0	0
Plug-in: Shockwave Flash	8,720K	5	N/A

At the bottom of the window, there is a link 'Stats for nerds' and a button 'End process'.

For all of these reasons, Google Chrome's multi-process architecture can help it be more robust, responsive, and secure than single process browsers.

## 2. What goes in each process?

Google Chrome creates three different types of processes: browser, renderers, and plug-ins.

*Browser.* There's only one browser process, which manages the tabs, windows, and "chrome" of the browser. This process also handles all interactions with the disk, network, user input, and display, but it makes no attempt to parse or render any content from the web.

*Renderers.* The browser process creates many renderer processes, each responsible for rendering web pages. The renderer processes contain all the complex logic for handling HTML, JavaScript, CSS, images, and so on. We achieve this using the open source WebKit rendering engine, which is also used by Apple's Safari web browser.

Each renderer process is run in a sandbox, which means it has almost no direct access to your disk, network, or display. All interactions with web apps, including user input events and screen painting, must go through the browser process. This lets the browser process monitor the renderers for suspicious activity, killing them if it suspects an exploit has occurred.

*Plug-ins.* The browser process also creates one process for each type of plug-in that is in use, such as Flash, Quicktime, or Adobe Reader. These processes just contain the plug-ins themselves, along with some glue code to let them interact with the browser and renderers.

## 3. When should the browser create processes?

Once Google Chrome has created its browser process, it will generally create one renderer process for each instance of a web site you visit. This approach aims to keep pages from different web sites isolated from each other.

You can think of this as using a different process for each tab in the browser, but allowing two tabs to share a process if they are related to each other and are showing the same site. For example, if one tab opens another tab using JavaScript, or if you open a link to the same site in a new tab, the tabs will share a renderer process. This lets the pages in these tabs communicate via JavaScript and share cached objects. Conversely, if you type the URL of a different site into the location bar of a tab, we will swap in a new renderer process for the tab.

Compatibility with existing web pages is important for us. For this reason, we define a web site as a registered domain name, like google.com or bbc.co.uk. This means we'll consider sub-domains like mail.google.com and maps.google.com as part of the same site. This is necessary because there are cases where tabs from different sub-domains may try to communicate with each other via JavaScript, so we want to keep them in the same renderer process.

There are a few caveats to this basic approach, however. Your computer would start to slow down if we created too many processes, so we place a limit on the number of renderer processes that we create (20 in most cases). Once we hit this limit, we'll start re-using the existing renderer processes for new tabs. Thus, it's possible that the same renderer process could be used for more than one web site. We also don't yet put cross-site frames in their own processes, and we don't yet swap a tab's renderer process for all types of cross-site navigations. So far, we only swap a tab's process for navigations via the browser's "chrome," like the location bar or bookmarks. Despite these caveats, Google Chrome will generally keep instances of different web sites isolated from each other in common usage.

For each type of plug-in, Google Chrome will create a plug-in process when you first visit a page that uses it. A short time after you close all pages using a particular plug-in, we will destroy its process.

We'll post future blog entries as we refine our policies for creating and swapping among renderer processes. In the mean time, we hope you see some of the benefits

# of a multi-process architecture when using Google Chrome.

Posted by Charlie Reis



[Google](#) · [Privacy](#) · [Terms](#)