深入了解现代网络浏览器（第 1 部分）

Mariko Kosaka

CPU、GPU、内存和多进程架构

在本系列博文（共 4 部分）中，我们将从高级架构到高级架构，全面介绍 Chrome 浏览器 渲染流水线的具体细节如果您想知道浏览器是如何将您的代码 正常运转的网站，或者您不确定为什么建议使用特定的技术来提高效果 本系列视频就是您的理想之选。

作为本系列的第 1 部分，我们将介绍核心计算术语，以及 Chrome 的 多进程架构

★ **注意**：如果您熟悉 CPU/GPU 的概念， 和进程/线程 浏览器架构。

计算机的核心是 CPU 和 GPU

要了解浏览器的运行环境，我们需要了解 计算机零部件及其用途

CPU

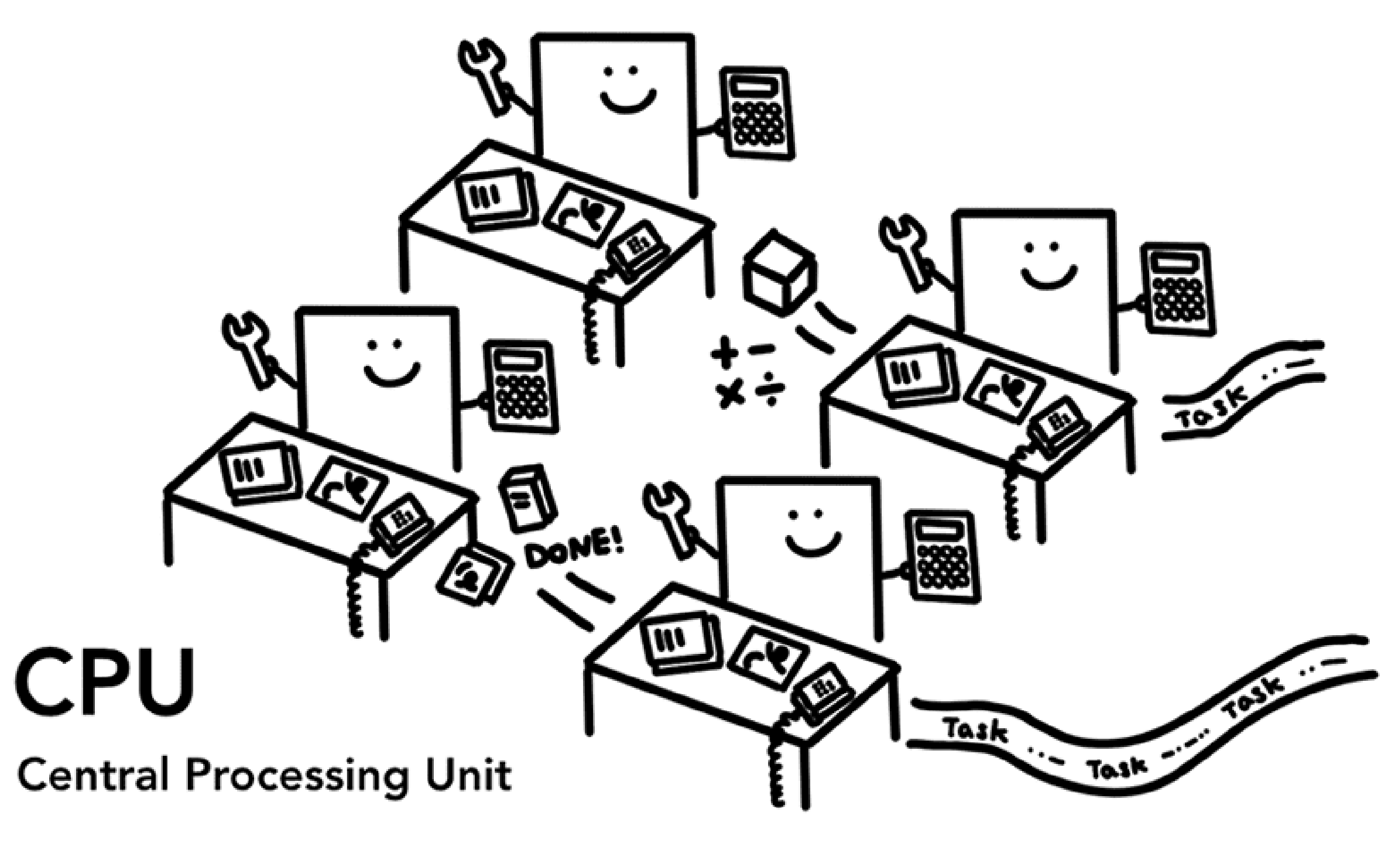

图 1：办公人员坐在各自的办公桌前处理传入任务的 4 个 CPU 核心

第一个是中心**处理**函数 (CPU)。您可以将 CPU 视为 计算机大脑。CPU 核心（如图中描述的办公室员工）可以处理许多不同的任务 逐一查看这些状态信息它能处理数学、美术等各种任务，而且知道如何回复 。过去，大多数 CPU 都是单芯片。核心就像是生活在 同一条状标签。在现代硬件中，您通常会获得多个核心，从而提供更好的计算能力 添加到手机和笔记本电脑上。

GPU

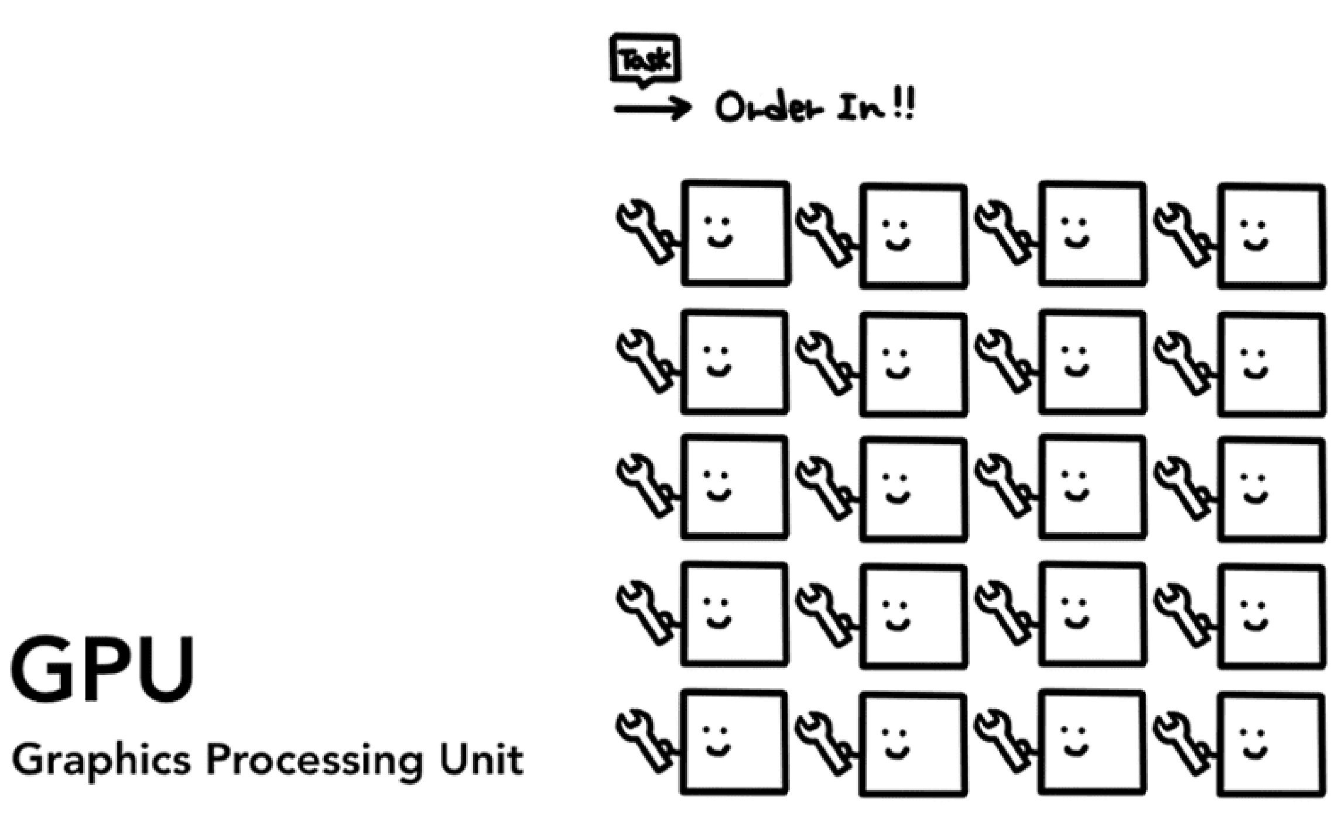

图 2：许多带扳手的 GPU 核心建议它们处理有限的任务

图像处理与 CPU 不同 GPU 擅长处理简单的任务，但可以同时跨多个核心。作为名称 表明，它最初是为了处理图形而开发的。这就是为什么 "使用 GPU"或“支持 GPU”与快速渲染和流畅交互有关。 近年来，随着 GPU 加速计算的推出，越来越多的计算可以在 仅使用 GPU。

当您在计算机或手机上启动应用时，CPU 和 GPU 是驱动 应用。通常，应用使用 操作系统。

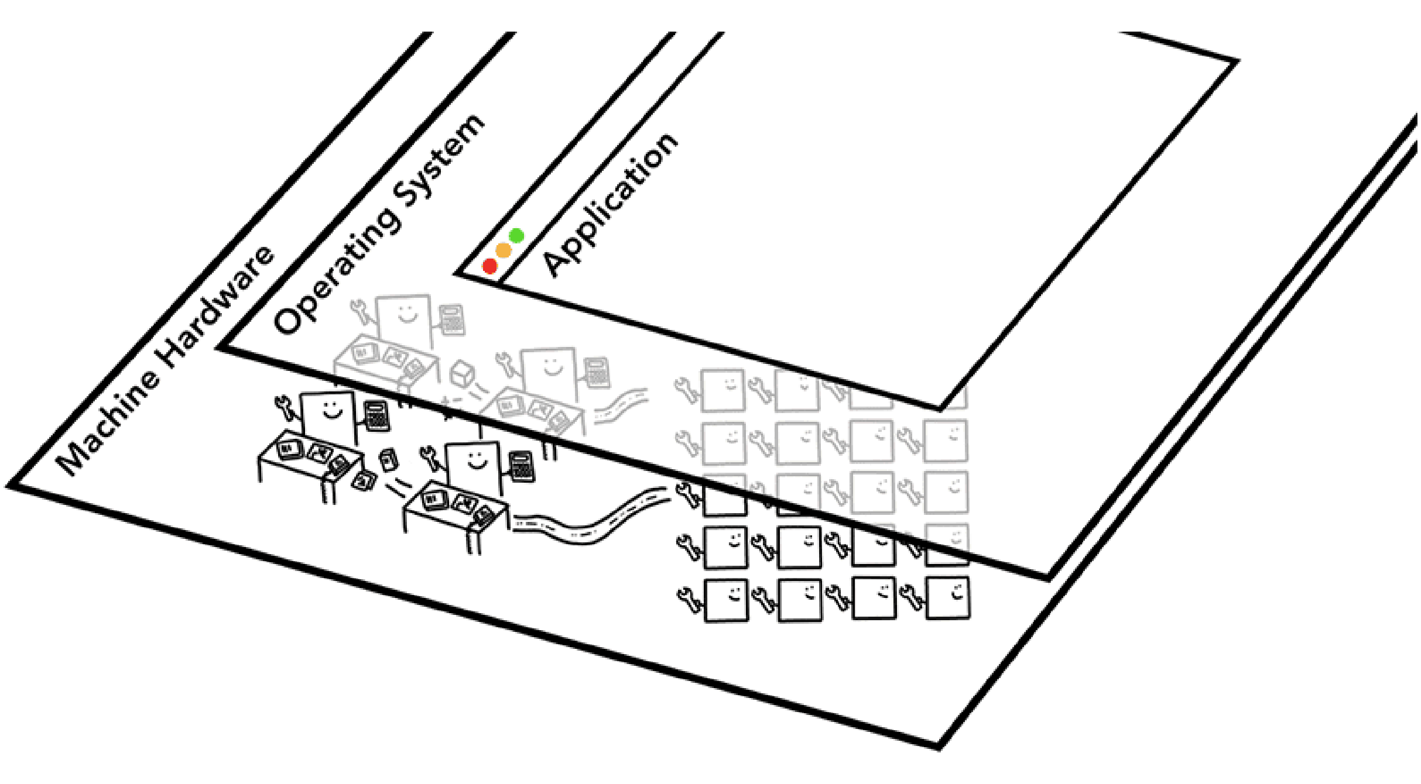

图 3：计算机架构的三层。底部的机器硬件，运行中 中间为“系统”，顶部为“应用”。

在进程和线程上执行程序

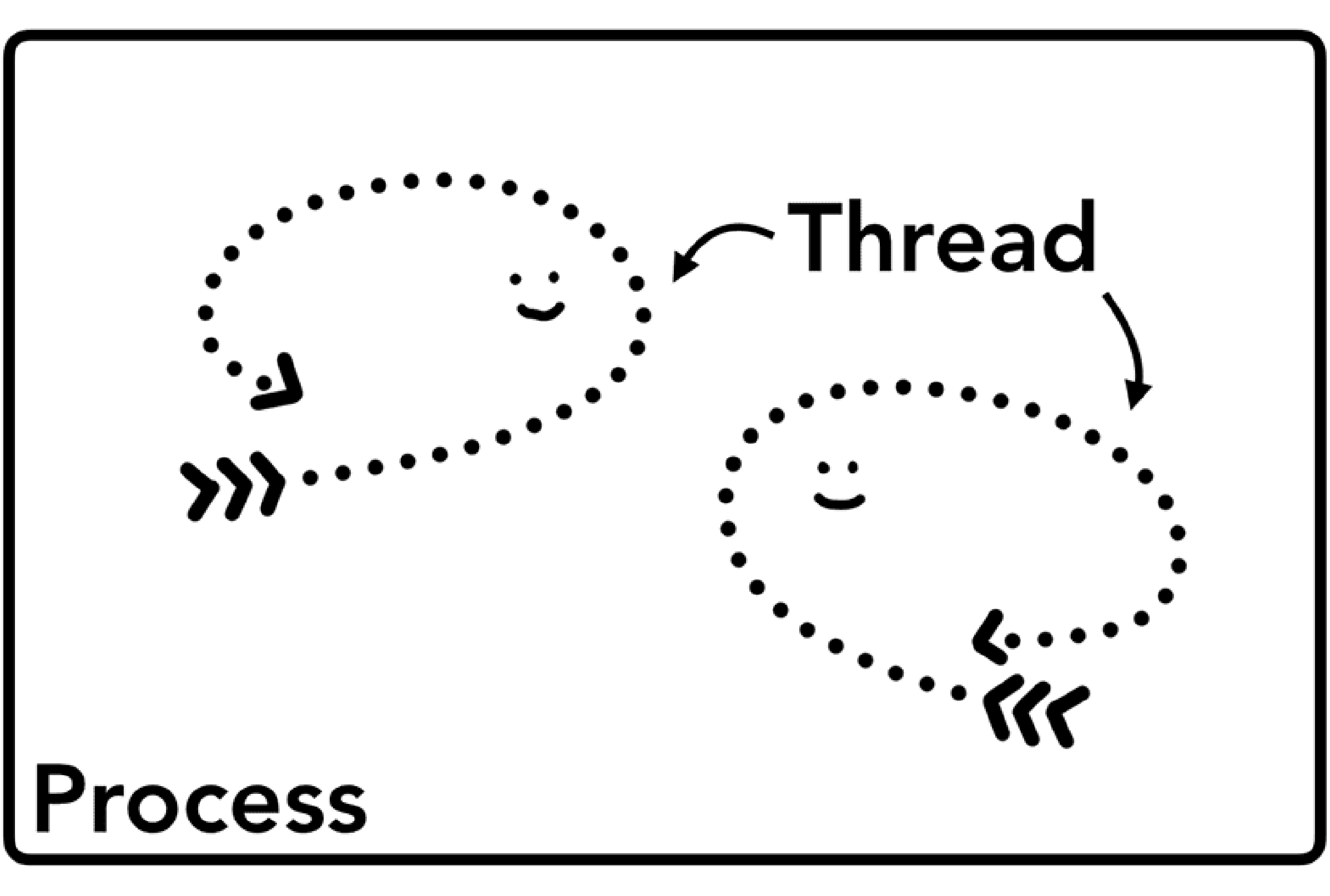

图 4：作为边界框的进程，即以抽象鱼在进程中游动的线程

在深入了解浏览器架构之前，另一个需要掌握的概念是进程和线程。 进程可以描述为应用的执行程序。线程是存在于 并执行其进程的任何部分。

启动应用时，系统会创建一个进程。程序可能会创建线程来帮助它 可以运行，但这是可选操作。操作系统为这一过程提供了一个“平台”运行 并且所有应用状态都保存在该私有内存空间中关闭 该进程也会停止，操作系统也会释放内存。

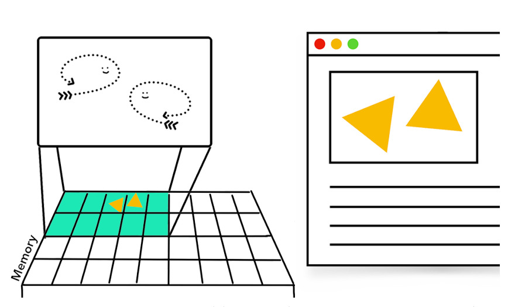

图 5：使用内存空间并存储应用数据的进程示意图

进程可要求操作系统启动另一个进程以运行不同的任务。当 内存的不同部分会分配给新进程如果两个进程需要 谈话时，他们可以使用私密**通信** (**IPC**) 来实现。许多应用 按照这种方式工作，如果工作器进程无响应，可以将其重启 而无需停止运行应用不同部分的其他进程。

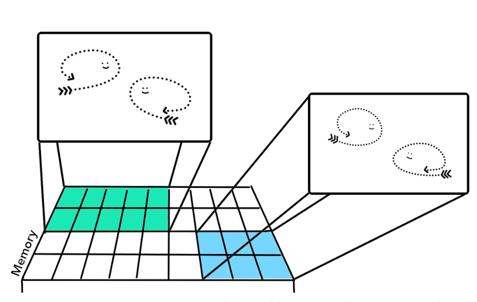

图 6：通过 IPC 进行通信的独立进程示意图

浏览器架构

那么，如何使用进程和线程构建网络浏览器呢？这个流程可以涉及许多 不同线程或许多不同进程与几个通过 IPC 进行通信的线程。

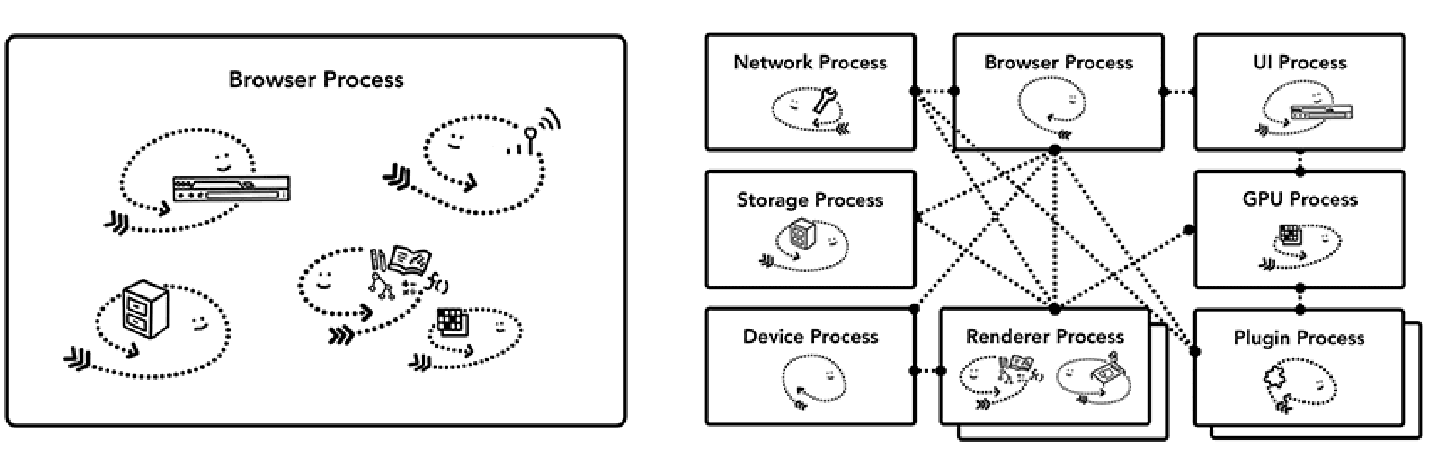

图 7：进程 / 线程中的不同浏览器架构图

这里需要注意的重要一点是，这些不同的架构是实现细节。 构建网络浏览器并没有一个标准规范。一种浏览器处理方式完全不同

在本系列博文中，我们将使用 Chrome 浏览器的最新架构， 如图 8 所示。

顶层是浏览器进程与负责不同服务的其他进程协调 各个部分对于渲染程序进程，系统会创建多个进程， 。直到最近，Chrome 浏览器都尽可能地为每个标签页提供了一个进程；现在它试图 每个网站都有自己的进程，包括 iframe（请参阅网站隔离）。

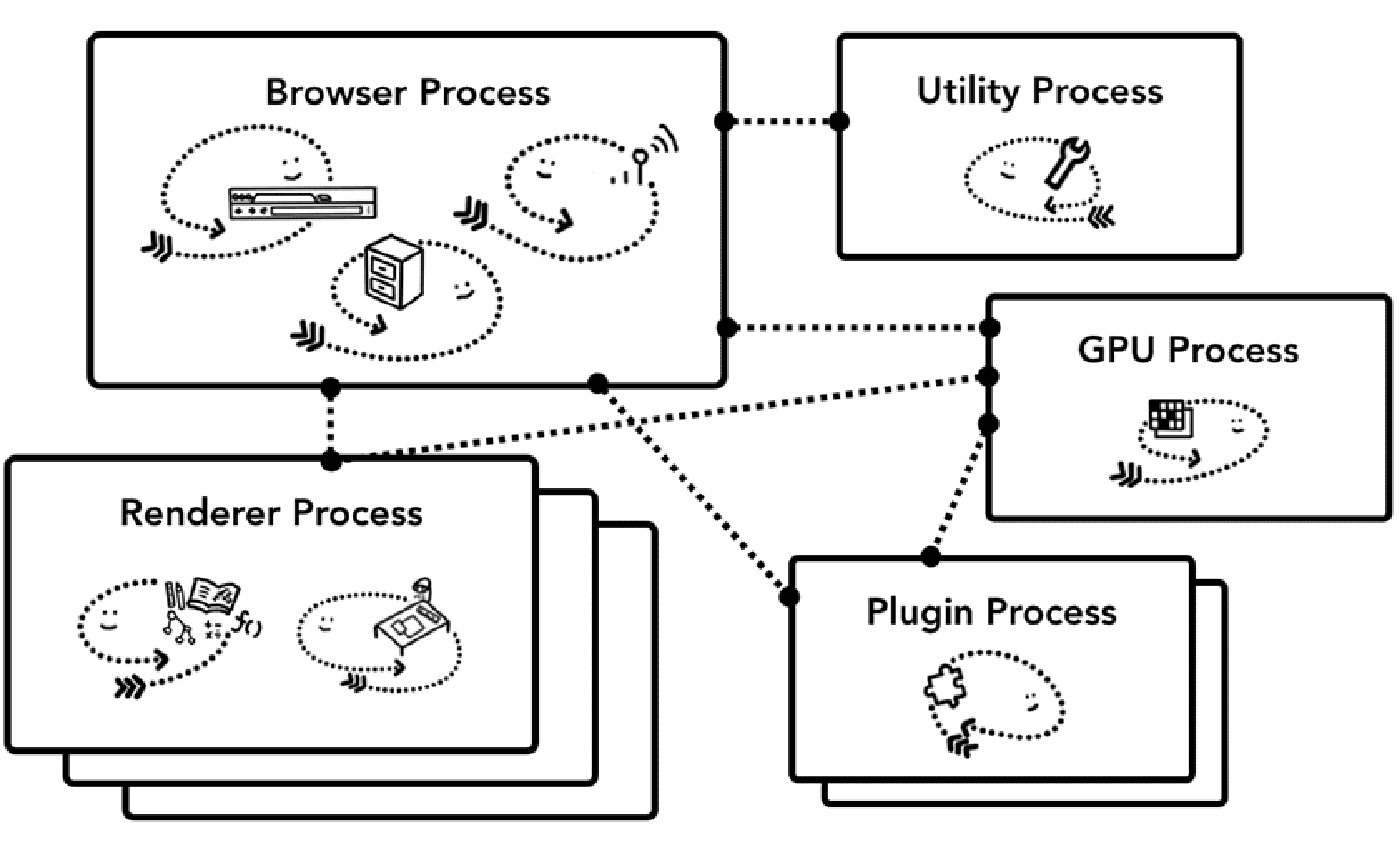

图 8：Chrome 的多进程架构图。下面显示了多个图层 渲染程序进程，用于表示为每个标签页运行多个渲染程序进程的 Chrome。

哪个流程控制什么？

下表介绍了每个 Chrome 进程及其控制的内容：

流程及其控制的内容	
浏览器	控制“Chrome”包括地址栏、书签、返回 前进按钮。 还可以处理网络浏览器中不可见的特权部分，例如 网络请求和文件访问
渲染程序	控制标签页内显示网站的一切内容。
插件	控制网站使用的所有插件，例如 Flash。
GPU	与其他进程分开处理 GPU 任务。它分为不同的进程 因为 GPU 会处理来自多个应用的请求，并在同一 Surface 上绘制它们。

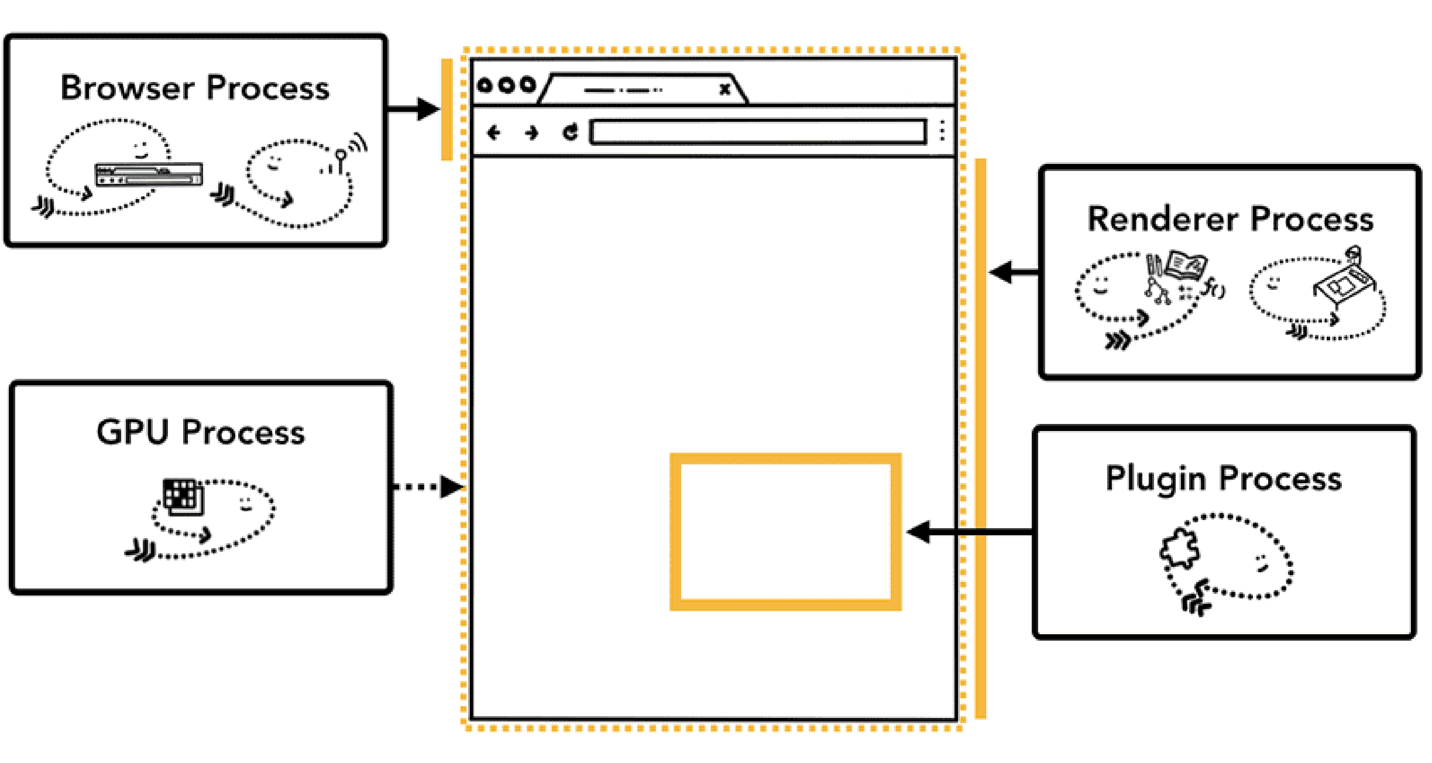

图 9：指向浏览器界面不同部分的不同进程

还有一些进程，例如 Extension 进程和实用程序进程。如果您想查看 请点击“选项”菜单图标 ⋮ ，选择“更多工具”，然后选择 选择“任务管理器”系统会打开一个窗口，列出当前正在运行的进程 以及它们的 CPU/内存用量

Chrome 中多进程架构的优势

我之前提到过 Chrome 浏览器使用多个渲染程序进程。在最简单的情况下，您可以 假设每个标签页都有自己的渲染器进程。假设您打开了 3 个标签页 由独立的渲染程序进程决定

如果某个标签页无响应，您可以关闭无响应的标签页并继续操作，同时保持 其他标签页处于活动状态如果所有标签页都在一个进程中运行，那么当一个标签页无响应时，所有标签页 标签页无响应太可惜了。

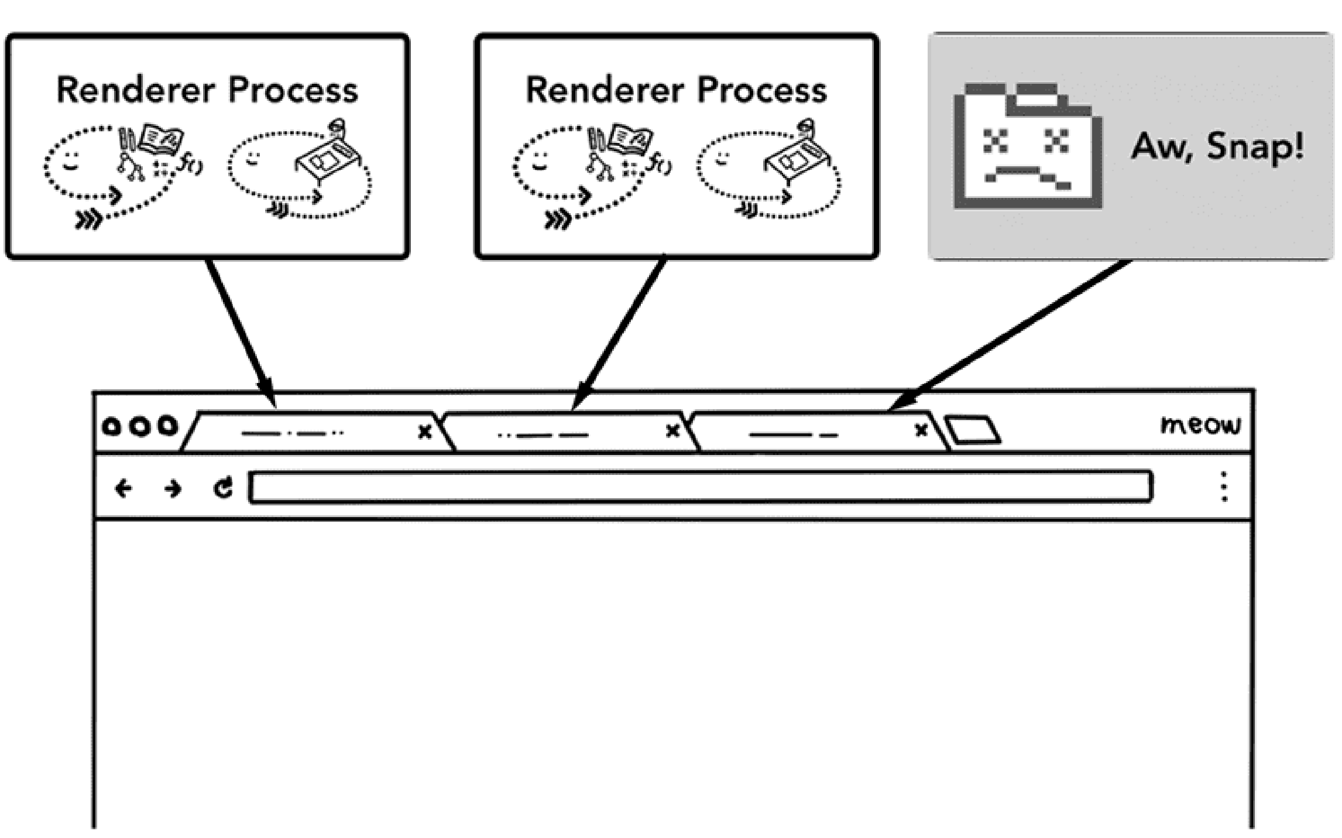

图 10：显示运行每个标签页的多个进程的示意图

将浏览器工作分成多个进程的另一个好处是，既能确保安全性， 沙盒由于操作系统提供了一种方法来限制进程的浏览器权限 可以将某些进程从特定功能沙盒化。例如，Chrome 浏览器 对处理任意用户输入的进程（如渲染程序进程）使用任意文件访问权限。

由于进程拥有自己的私有内存空间，因此通常包含 （如 V8 - Chrome 的 JavaScript 引擎）。这意味着内存用量会随着 如果它们是同一进程中的线程，则无法像共享它们那样共享它们。 为了节省内存，Chrome 对可以启动的进程数量设有限制。 该限制因设备的内存和 CPU 性能而异，但当 Chrome 命中时 即开始在一个进程中运行来自同一网站的多个标签页。

节省更多内存 - Chrome 中的服务功能

浏览器进程也采用同样的方法。Chrome 正在进行架构更改 将浏览器程序的各部分作为服务运行，允许拆分为不同的进程 也可以将其合并为一个应用

一般来说，当 Chrome 在功能强大的硬件上运行时，可能会将每项服务拆分为 不同的进程赋予更高的稳定性，但如果在资源受限的设备上，则 Chrome 将服务整合到一个进程中，从而节省内存。类似的整合方法 在此次变更之前，Android 等平台上都使用了内存使用量较少的进程。

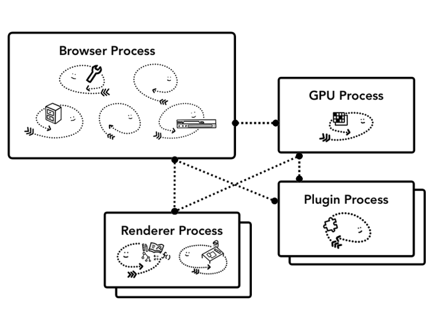

图 11：Chrome 服务将不同服务移到多个进程中的示意图 和单个浏览器进程

每帧渲染程序进程 - 网站隔离

网站隔离是最近 在 Chrome 中引入了一项功能，可为每个跨网站 iframe 运行单独的渲染程序进程。 我们一直在介绍每个标签页模型一个渲染程序进程，该模式允许跨网站 在单个渲染器进程中运行的 iframe，不同网站之间共享内存空间。 在同一个渲染器进程中运行 a.com 和 b.com 似乎没有问题。 同源政策 是网络的核心安全模型；可确保一个网站无法访问来自其他网站的数据 未经同意。绕过此政策是安全攻击的主要目标。 进程隔离是分隔网站最有效的方法。包含 Meltdown 和 Spectre， 更明显的是，我们需要使用流程来分隔网站。 从 Chrome 67 开始，在桌面设备上默认启用网站隔离功能后，标签页中的每个跨网站 iframe 会获得一个单独的渲染程序进程

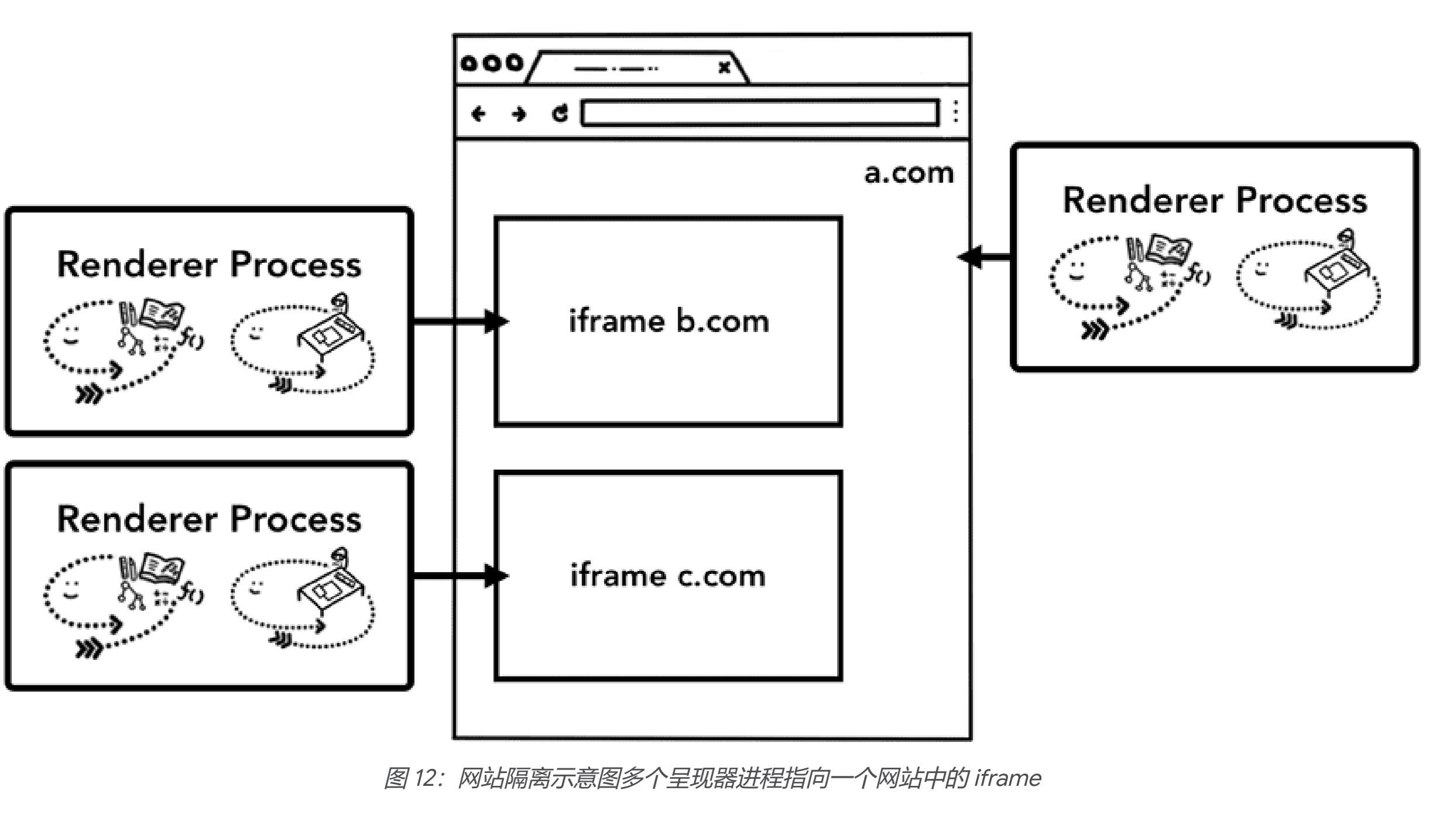

图 12：网站隔离示意图多个呈现器进程指向一个网站中的 iframe

启用网站隔离是一项多年来的工程工作。网站隔离不像 分配不同的渲染器进程；它从根本上改变了 iframe 与每个 其他。如果某个网页上的 iframe 在不同进程上运行，那么在该网页上打开开发者工具意味着开发者工具必须 实施幕后工作，使其看起来无缝衔接。甚至可以使用简单的 Ctrl+F 表示在不同的渲染器进程中进行搜索。您可以看到 各位浏览器工程师谈论，网站隔离功能的发布已成为一个重大里程碑！

小结

在这篇博文中，我们概要介绍了浏览器架构，并介绍了 多进程架构我们还介绍了 Chrome 中的服务和网站隔离， 与多进程架构密切相关。在下一篇帖子中，我们将深入探讨 在这些进程和线程之间发生，以便显示网站。

您喜欢这个帖子吗？如果您对以后的帖子有任何疑问或建议， 通过 Twitter 向 @kosamari 发送最新动态。

下一页：导航过程中会发生什么