我们刚刚吃完玉米饼(tacos)（星期二(Tuesday)+ 苹果事件(AppleEvent) = 星期二在 CodeWeavers 的玉米饼事件），因为对 2019 苹果发布会充满了期待。我们的手指交叉着，紧握着我们选择的幸运符（四叶草、马蹄形符、毛绒骰子、兔脚钥匙链和捕梦网），我们多么希望能有一个线索可以知道最新的 macOS Catalina 何时会发布。

然而我们一无所获。除了在苹果发布会后更新了他们的网站，含糊提到 10 月份将发布一个版本。好吧，至少这给我们争取到的时间比九月的任何一个星期二都要多。

我们还没有关于 Catalina 的任何信息，但是我们有一个关于技术曲线球的解释，我们已经取得的进展，我们认为适合我们发布与 CrossOver 兼容的 Catalina 的时间，我们的支持团队对您的指导，以及当我们的产品发布时在 Catalina 上不起作用时的善意姿态。

>>>**>>>若要了解 Catalina 的最新进展，请加入我们的 "mac OS Catalina Update" 邮件列表**

技术曲线球

作者：Ken Thomases

随着 macOS 10.15 (Catalina) 的发布，苹果放弃了对运行 32 位可执行文件的支持，并删除了 32 位版本的系统框架和库。我们的用户所使用的大多数 CrossOver Windows 应用程序都是 32 位的，CrossOver 使用 32 位 Mac 可执行文件、系统框架和库来运行这些应用程序。这与 Catalina 背道而驰。（即使 64 位 Windows 应用程序通常也有 32 位安装程序。）

注意位数

---

32 位和 64 位业务到底是怎么回事？“位数”是 CPU 体系结构的一个元素。它说明 CPU 可以“自然”和有效处理的规格数字。通过在硬件能力的基础上进行构建，软件可以处理任意大的数字，但硬件本身只能直接处理指定的位数或更小的位数。

位数还决定了硬件可以直接访问多少内存。32 位的 CPU 最多可以直接访问 4GB 的内存。64 位的 CPU 最多可以访问 256TB（约为 65,000 倍）。（理论上，64 位代表 40 亿倍，但 CPU 实际上还不支持这么大的规格。）

多年来，个人计算机已经从 8 位发展到 16 位、32 位，现在是 64 位。在 Mac 中使用的基于 Intel 的 CPU 系列中，每当位数增加时，硬件都保持一定程度的向后兼容性，以便现有的软件能够继续运行。然而，操作系统 (OS) 必须管理和支持 CPU 的不同模式，以保持这种向后兼容性。

应用软件是为特定的位数而构建的。因此，有些软件是 32 位的，而其他软件是 64 位的。在底层，软件只是一系列字节（小数字）。CPU 将这些字节解释为指令。从字节到指令的转换在 32 位和 64 位之间类似，但不相同。而且，在计算中，仅仅接近是不够的。如果试图将 32 位指令编码的字节序列解释为 64 位指令，或者进行相反的操作，则将失败。程序几乎马上就会崩溃。

当 64 位 CPU 需要运行 32 位软件时，操作系统将其切换到 32 位兼容模式，以便知道如何将该软件的字节解释为 32 位指令。不仅如此，CPU 的行为方式必须与 32 位 CPU 的行为方式相同，这样指令才能产生预期的结果。

当 CPU 需要运行 64 位软件时，操作系统将其切换到 64 位模式。在此模式下，它将指令字节解释为 64 位指令，并支持 64 位提供的更大功能。

应用软件不仅仅需要与 CPU 交互。现代操作系统的许多功能以系统框架和库的形式提供给应用程序。这包括在屏幕上显示窗口和图形，接受来自键盘和鼠标的用户输入，播放声音等功能。32 位应用程序需要 32 位系统框架和库；64 位应用程序需要 64 位系统框架和库。

于是，问题来了。

Catalina 无法运行 32 位可执行文件。它不支持 32 位进程。它不提供 32 位系统框架和库。简而言之，它无法运行 32 位组件，即 CrossOver 传统上用于运行 32 位 Windows 应用程序的组件。

所以，我们需要找到另一种方法。我们需要在 64 位进程中运行 32 位代码，该进程使用 64 位系统库来访问 OS 功能。

Mac 中的 CPU 仍然支持 32 位兼容模式。Catalina 仍然为我们提供了一种方法，使我们可以在 32 位模式下在 64 位进程中运行某些特定代码。我们面临的一个挑战是，当在两种类型的代码之间切换时，如何根据需要来回切换模式。

另一个挑战是在 64 位进程中维护 32 位进程可以处理的环境。例如，32 位代码需要与之交互的所有代码和数据必须位于内存的底部 4GB 内。回想一下，32 位代码最多只能访问 4GB 的内存。

相对地，内存地址（“指针”）在 Windows 应用程序和 Windows OS 之间的接口中经常使用。由于 CrossOver 用于代替 Windows OS，它需要与 Windows 具有系统的接口和 Windows 应用程序保持一致。对于 32 位软件，指针的大小为 32 位。对于 64 位软件，指针的大小为 64 位。我们需要找到一种方法来编写既能理解 32 位指针（用于与 Windows 应用程序对接）又能理解 64 位指针（用于与系统库对接）的软件。

进展

正在研究解决方案

CrossOver 用于在 macOS 上运行 Windows 应用程序的技术被称为 Wine。它是一个大型的、复杂的软件。克服这些挑战所需的改变将需要改变整个 Wine 的行为。如果我们试图手动完成这项工作，那将是一项非常艰巨的任务。

为了大幅降低手动完成的比例，我们依赖于一个可以控制 Wine 行为核心的工具：编译器。编译器是软件开发人员用来将源代码转换为 CPU 可以直接执行的指令字节序列的工具。我们构建了一个 macOS 标准 C 语言编译器 Clang 的修改版，以便在不对 Wine 的源代码进行普遍更改的情况下，自动化我们需要对 Wine 的行为所做的许多更改。

首先，我们的 Clang 版本理解 32 位和 64 位指针。我们能够从一个宽泛的层次到一个详细的层次来控制 Wine 源代码中哪些指针需要是 32 位的，哪些是 64 位的。在与 Windows 应用程序对接的接口上，任何替代 Windows 的代码都必须使用 32 位指针。另一方面，系统库的接口始终是 64 位的。

Wine 中有一些地方，它从系统库接收指向数据的指针，并需要将该数据传递给 Windows 代码。这是难题。Windows 代码将无法处理从系统库返回的 64 位指针。如果我们试图按原样传递指针，64 位指针的前 32 位就会被截断，导致指针指向错误的位置。

因此，我们必须手动处理这些问题。但首先我们得找到他们！如果仔细研究代码并考虑将一个指针复制到另一个指针的每个位置，则会花费太长的时间，并且很容易出错。为此，我们的 Clang 版本将在编译期间将代码中的某些地方标记为错误，迫使我们修复这些错误，但可以确保其他的代码是正确的。

接下来，Clang 需要识别代码中 64 位 Wine 代码可能需要调用 32 位 Windows 代码的地方，反之亦然。它需要以平滑转换的方式编译我们的代码。已对“普通”的 Wine 进行了一些上述操作，但需要更广泛的操作。我们的 Clang 生成 "thunk"，这是一小段代码，用于调解其他两段代码之间的转换。当 Wine 的 64 位代码需要从 Windows 应用程序调用 32 位代码时，Clang 会生成 64 位到 32 位的 thunk。当 Wine 系统需要为 32 位 Windows 代码提供一个入口点来调用它认为是 32 位的 Windows 函数，但实际上在 Wine 中实现为 64 位函数时，Clang 会生成 32 位到 64 位的 thunk。这些 thunk 管理 CPU 两种模式之间的切换。

借助这些工具（和其他一些工具），我们已经能够取得良好的进展！但是，不幸的是，当发布 Catalina 时，似乎还未完全准备就绪。工作还在继续！

时间线

作者：Ulrich Czekalla 和 James Ramey

2017 年，开始

2017 年年中，苹果首次暗示，在某个即将发布的 macOS 版本中，它将终止/放弃对 32 位的支持。乍一看，软件开发者似乎得到了充分的警告，他们的 32 位软件应用程序将停止工作，需要采取行动。虽然这在很大程度上是正确的，但它并没有描绘出实际时间线的准确画面。虽然苹果公司在 2017 年宣布了这一消息，但直到 2018 年 9 月 24 日发布了 Mojave (macOS 10.14)，时钟才开始滴答作响。随着 Mojave 的发布（实际上是 Mojave 在 6 月初的初始测试版），我们的开发团队更加清楚对 32 位的支持将如何逐步淘汰，对 64 位的支持将如何充分实现，以及 CodeWeavers 为了保持兼容性必须在 CrossOver19 中满足的基本要求。请记住，将 CrossOver 本身更新为 64 位并不是问题所在。我们面临的挑战是，在不再支持 32 位代码的操作系统上继续执行 32 位 Windows 应用程序，并以在 macOS 上维护现有 Windows 应用程序的无缝执行的方式执行这些应用程序。从那一刻起，我们组建了一个团队来设计和实现跨界解决方案（大约 14 个月前）。

2018 年，中期

在没有拿到任何 Catalina (macOS 10.15) 构建的情况下，我们的开发团队不得不考虑如何保持 32 位支持。在根据最可能的限制因素确定最佳可用解决方案方面花费了相当大的脑力。从理论上讲，完成将来的工作似乎是相当可能的；然而，团队只能假设 64 位操作系统上的 32 位虚拟化支持将如何影响应用程序性能，或者单个类型的解决方案如何支持使用 32 位启动器的 64 位应用程序。由于问题多于答案，我们团队只能推测其他可能的问题。

2019 年，结束？

快进到 2019 年 6 月 24 日，苹果公司宣布并发布了 Catalina 的第一个公开测试版 (macOS 10.15)。随着 Catalina 第一个公共测试版的发布，我们的开发团队终于能够测试理论并评估 CrossOver 19 软件的状态。不出所料，CrossOver 19 在 Catalina 上严重受损。从那一刻到今天，我们的团队已经着手实施基线兼容性所需的“管道系统（plumbing）”。在此期间，他们在使 32 位测试应用程序 (NotePad) 在 CrossOver19 中正常运行方面取得了重大进展。他们还确定了其他需要解决的关键问题，以便为 32 位应用程序提供更广泛的支持。而且（最重要的是）该团队已经将 CrossOver 19 升级到可以在其他苹果电脑上安装和运行并获得类似结果（如果不是相同的话）的版本。所有这些对于最快地发现和修复 bug 以及将 CrossOver 19 转换为 alpha/beta 版本都是至关重要的。

我们的当前目标是在未来的 30-60 天内为客户提供 Crossover 19 的 alpha 版本。理想情况下，通过我们的加速努力，我们可以更快地交付可用的软件；然而，在 64 位操作系统上支持 32 位应用程序是一项艰巨的挑战（只有 CodeWeavers 才会尝试这样做）。我们的开发团队拥有极高的专业才能和执着精神；但重申一遍，这是其他公司很少能解决的技术挑战。虽然两年（理论上）解决这样一个问题似乎是可行的，但现实情况是，我们的开发团队实际上只有一年的时间来开发解决方案，而与 Catalina 一起开发和实施解决方案的时间只有几个月。

指导方针

作者：Ryan Abhiram 和 Brian Mathieu

在 CrossOver 19 发布之前，Windows 应用程序将无法在 macOS 10.15 上运行。如果您看到一个提示您更新到 macOS Catalina 的通知，您可以简单地关闭该消息，并且一切将继续运行，就像它当前所做的那样。

如果已经更新到 macOS Catalina 或意外地完成了更新，您有两个选择。第一种选择是等待 CrossOver 19 发布，并且不使用 CrossOver 中的任何 Windows 应用程序。这不是大多数用户的首选。

第二个选择是回退到升级 macOS 之前的一个时间点。如果在安装 Catalina 之前已经有“时间机器”备份，则可以重新安装系统并从备份中还原。请注意，这将清除 macOS 更新后生成的任何数据以及对系统所做的任何更改。有关此操作的详细说明，请访问 http://support.apple.com/guide/mac-help/revert-to-a-previous-macos-version-mh15216/mac

如果您还没有备份，仍然有希望。可以附加外部硬盘驱动器并创建第一个“时间机器”备份。备份完成后，请与苹果公司联系以获得安装早期版本的 macOS 和手动迁移信息的帮助。没有自动执行此操作的工具，因此您需要确保有天才吧或苹果电话支持 (400-666-880) 的指导。

商誉

作者：Jana Schmid

事情是这样的。我们是好人。如果我们的产品暂时不能工作，我们认为您不应该为它买单。从 2019 年 9 月 10 日起，任何拥有 Crossover macOS 许可的用户都将自动获得免费的三个月额外支持，直到我们可以在让 CrossOver 19 在 Catalina 上正常运行。如果我们需要的时间，您将额外获得更多。如果我们在两个月内启动并运行 CrossOver 19，您仍将获得三个月的额外支持。

例如：Bruce 的一年 CrossOver 许可将于 2019 年 9 月 30 日到期。因为 Catalina 的 CrossOver 19 还未完全准备就绪，CodeWeavers 给他增加了三个月的订购时间。Bruce 的许可将于 2019 年 12 月 30 日到期。Bruce 很开心，晚上睡得很好，因为他知道自己在支持开源社区。