Windows Phone | Windows 10 | 通用 Windows 平台（UWP）

关注者 609 | 被浏览 43,397

Windows 10 通用应用的前景如何?

Win10 Mobile 想借桌面的霸主地位，利用通用架构改善生态质量，大家感觉前途怎么样?

关注问题 写回答 2 条评论 分享 邀请回答 举报 ...

42 个回答

默认排序

林行书
当我在装逼

361 人赞同了该回答

本来我不想在zhihu说什么话的，就是围观各路大神装逼就好了，但是作为一个对Win32有着深厚感情的程序员，**我必需出来反击**一下@曹一聪。
首先表明一下立场，**我并不反对UWP**，也不会说UWP现在、未来如何，我只是想说一下，Win32是什么，Win32和UWP\Appx是什么关系，也请了解后的人，**不要对Win32、桌面exe抱有那么大的偏见**，因为这些是伴随你我青春和成长的东西，虽然以后它可能无法伴随很多年轻人的青春了。
我文章有点长，因为这也是我的一些回忆。

Win32是怎么来的呢，虽然NT之前就已经有Win32的概念(那叫做Win16，但是API今天依然有，比如经典的LocalAlloc和GlobalAlloc的区别，搜一下就知道了)，但是Win32的分层真正还是从Cutler做的NT开始的。
我们抛开啰嗦的内核层不谈，因为内核层有不确定性。

应用层所有Windows程序都会经过Win32\COM API进入到NTAPI层，然后由NTAPI层通过sysenter方式将调用分发到内核。
NTAPI层在桌面Windows有两套，一套是由ntdll做thunk的，分发到ntoskrnl.exe，我们叫它SSDT表；另一套是NT的User界面子系统，由user32或者gdi32做thunk，这套系统在NT4以前是使用应用层的csrss运行的，当然现在csrss也还是承担着一部分的绘图工作，比如命令行窗口，现在则把user32的Window函数，消息循环逻辑、部分gdi的绘图逻辑，都分发到了win32k.sys，你可以叫做影子SSDT表，为什么叫影子，因为当年这个表微软不导出，但是需要做一些窗口的保护驱动，需要这个表，然后大家都是搜内存搜出来的，所以就叫做影子。

在Windows Phone 8上或者今天的Windows 10 Mobile上（基于NT内核），微软就把win32k.sys删空了，自然user32、gdi32也不存在了，全部使用DX绘图，也就是把桌面的窗口、消息循环、gdi删掉了，只留下了DX的驱动，然后使用DX来绘制UI，这种绘制UI的方法，**用桌面程序员的话叫做DirectUI**，最早起源与微软控制面板的窗口类名，但是桌面程序员是不可能拿DX绘图桌面exe的UI的，因为要考虑大量XP用户，大量没装显卡驱动的用户。
你今天所使用的大部分桌面大型UI应用，比如QQ、迅雷、酷狗、360、金山，在那个Chrome做客户端UI还没有兴起的年代，他们的DirectUI大部分都是自己写的一套库，腾讯叫做GF，迅雷叫做Blot，新浪微薄的exe桌面客户端就是使用迅雷的blot。
（但是我印象没记错的话，WP8时代的Office软件是WP团队自己搞的，记得因为是从桌面移植了一些模块，所以微软的团队硬是用DX封装了一个GDI\GDI+的兼容层在Office里面，导出了兼容桌面GDI\GDI+的函数。）

360安全卫士在早期Hook了一个叫做**KiFastSystemCall**的系统调用，这是一个未导出的函数，隐藏在sysenter切内核的函数代码里面，每个版本的系统的offset和patch-code都不同。Hook这个方法后，360可以抓到所有的系统调用，不管你应用层做什么事情，你移动一下鼠标1px也好，都会先经过360的模块。现在360应该做得更狠了，但是我几年不研究windows内核了，也不太清楚了。
（后来金山也Hook了这个，但是金山姿势有问题，还有漏洞，被MJ0011在百度博客喷）
腾讯的TP直接接管了系统的调试中断，所以你会发现开LOL了**VS就不能调试了**，因为腾讯直接把本机调试禁止了，这个办法也是判断系统版本做特征码的，所以系统一更新，TP跟不上，游戏就跑不起来了。（腾讯的TP我记得有不少开发者都是看雪论坛的老人了）
QQ桌面exe版本为了防止那种注入式外挂、聊天记录监视器，现在也加了个QQProtect.sys，Hook了SSDT里面的一些进程方法，防止你打开它进程，但是只能防笨蛋而已。

相关问题

Windows 10 能挽救 Windows 手机吗? 32 个回答
Lumia 会通过 Windows 10 崛起吗? 10 个回答
Windows10家庭版如何关闭445端口? 7 个回答
你怎么看 Ubuntu touch 系统和 Windows 10 for Phone 系统的前途? 22 个回答
现在（据说已经过了Windows10的免费升级时间）从Windows8升级成了Windows10? 11 个回答

好了，我们说完应用层软件的故事，我们再来说下Windows Runtime和appx。
WinRT API，其实是一套高度抽象接口化的**COM API集**，它跟传统的Win32 COM并无两样，只是把GUID换成了HSTRING，根对象IUnknown又多了一个IInspectable，视觉上更友好了一点。
WinRT API也不是什么底层的API，**是一个基于Win32之上的高层API**，也不是什么新技术，它真正创新的部分就是XAML和沙箱应用程序模型。
1、它的Async、ThreadPool逻辑基于Win32\NTAPI线程池，这套线程池在UWP也可以使用，起源于Vista系统。
2、它的大部分API都是**把老旧的Win32 API进行二次封装**，暴露为在ThreadPool中运行的所谓Async方法，比如HttpClient封装WININET，StreamSocket封装WS2_32，Devices使用RPC方式封装SETUPAPI，后台下载使用RPC方式封装BITS，USB封装WinUsb，Sensors封装Win7那套COM的传感器API，HumanInterfaceDevice封装HID.dll和一些RPC调用，证书封装老的CERT系列API，Windows.Media封装MediaFoundation等。
大部分Win32 API在Vista、Win7甚至XP就有了，比如uwp现在常用的HttpClient的内核WININET，Win2000时代(其实更早)的产物。
3、winmd就是COM的**typelib的C#版本**，因为typelib当年做出来是给VC和VB还有ASP中的COM自动化进行交互的，20世纪后.NET战略起步，一开始微软团队割裂严重，typelib这种老技术不适合.NET的优雅，.NET团队要实现C#的优美又要兼顾Windows的基石COM，所以他们只能把CLR运行时做成COM的来兼容一些老东西比如typelib，是的你没听错，**CLR在Windows上就是基于COM的**，关键字叫做CLRCreateInstance。这种COM和.NET割裂的情况一直持续到WPF，微软发现这样不行，他们C++和Windows团队要继续开发COM，但是.NET又深受其害，以至于出现很多COM的PInvoke封装品比如SharpDX这种东西，但是WinRT API又不能离开COM，怎么办，嘿嘿，所以在WinRT搞了个winmd，然后再搞一个不伦不类的语言C++\CX，表面上看CX貌似没COM，其实都是在生成一大堆COM，详细可以去看CX项目主管写的blog文章。
4、WinRT API为了权限和安全，把大部分调用，比如文件访问，**通过RPC**（远程过程调用），RPC在本机上实现为ALPC，ALPC把调用分发到一个高权限的进程，高权限的进程使用RmAccessCheck去检查客户端的Caps，判断这个客户端是否有权限访问，有的话高权限的进程去打开文件，然后把句柄再复制回去客户端进程，ALPC调用返回。正因为这繁琐的调用机制，所以在系统繁忙和Metro App同时运行过多的时候，会使Metro app的服务端（svchost.exe）负担过重，微软为了解决这个问题，就另做了一些单独的服务端，比如什么TimeBroker啊，SystemEventBroker什么的鬼之类的***Broker，打开文件时候的OpenFilePicker什么的，让他们单独跑一些负担过重的任务。
然后也整因为这种跨进程在应用层到处跑RPC的方式，到处都是安全lock和COM的内存列集和散集，使得某些WinRT API访问速度过慢，比如文件系统API，所以他们可能超过50ms，特别是在多个APP同时运行的情况下，所以微软把他做成了异步的。
在Win32桌面exe中，权限在启动的时候就已经在Token中写稳了，打开设备访问文件都是直接的，成功失败都是在内核决定的一瞬间的事情，所以效率很高。
5、XAML实现在应用层，使用DX渲染；沙箱应用程序模型实现在内核层和Win32层，大部分是Win32层的API在进行控制。
6、appx就是个压缩包，安装就是解压，读下里面的xml文件，把相关的权限信息写入注册表，APP激活的时候，LowBoxToken可以根据注册表里面的信息来创建。

也就是WinRT API本质是基于Win32\COM下的一套高层API，有WinRT就有Win32\COM，这是不可能分割开来的，就像Win2D基于D2D，XAML基于DX一样的道理。
关键是微软出于脑热，把一些Win32 API刻意禁止了，你可以hack来使用，但是用了可能上不去商店，**比如HttpClient基于的WININET，你可以使用HttpClient，但是你不能使用WININET**，微软的意思是你应该使用新的API，但是他自己就使用老的API，比如Edge、小娜、Win10的开始菜单都是C++和Win32、DX开发的，只是用XAML画了个UI。
很多Win10的新的系统库，使用的COM库还是ATL，而不是WRL，微软内部使用老的库很多，当然也是为了兼容桌面。
在Win10的C:\Windows\SystemApps下面，唯一一个真正的C#的UWP程序，就是那个Windows Feedback。

以后的趋势，假设UWP确实起来了，桌面轻量客户端转移UWP架构，那桌面客户端开发更多会使用WinRT API，毕竟XAML能带来触控更佳的体验，但是要更好的触控体验就需要牺牲功能，二者难同美，所以大型软件还是只能靠Win32。
还有WinRT API是一套**客户端趋向的API**，Win32是完善的企业、服务器的API，所以方向也不同，微软发那个不带WinRT的企业LTSB版本就是如此，**大部分企业不需要WinRT**。
二者在以后的UWP现代软件开发中，**是互补关系**，假设你开发过程中只学习WinRT不学习Win32，遇到棘手的问题你恐怕只能求救到那时候熊猫般的还会Win32的开发者了。
微软现在公开给UWP的Win32 API也是补全WinRT的，比如内核线程池、内核对象、音频视频、游

戏图形等，轻量级别的APP基本上不需要使用，但是不意味着可以没有。
假设没有WP8.0时代开放的那点Win32 API，那也没MoliPlayer。

还有A和W的问题。。。
恩，如果没系统性的开发过Win32，确实会纠结这个的，但是A和W是必需存在的，即便全UNICODE的今天，即便10年后Windows还活的话，A依然有，信不信我就立贴这里了。
（8.1时代微软还不开放A的API，UWP就开放了。
然后再反驳一下那句话：**Win32/COM那套API，是注定要被历史淘汰的。**
Edge的查克拉引擎，是一套C风格导出的Win32 API，里面封装了COM API。
真要淘汰Win32/COM，先把微软灭了吧。

编辑于 2016-01-17

王韦恩卑鄙
因为美术苦手 只好去做框架 萌大叔

114 人赞同了该回答

利益相关 微软UWP开发顾问。
应用的复杂度是和开发商的投入成正比的，所以吐槽应用简单的我只能摊手。

我说几个其他方案在Windows 上无法解决/很难解决的问题 这些UWP平台做到了。这些优点如何转化为用户可以感受到的价值，还是这句话 “应用的复杂度是和开发商的投入成正比的” 当然也“和开发人员和产品经理的愚蠢程度成反比”。我们一起努力吧。

1 省电。
UWP 开始引入了.net native,，没有Jit了编译在云上做了, 这是以前.net开发的程序无法企及的优势。可是大家现在没有对比，要么是app没出uwp,要么是app出了uwp ，但是会替换掉老app，同时开发者很可能改版，用户会以为是改版造成效果，所以这一效果没有什么人会有明显感觉。

2 启动速度快
同样是.net Native 带来的优势 ，同样代码从8.1 迁移到uwp启动快40%-60%是普遍现象

3 VR/AR的支持，
D3d开发的App 原生支持切换左右眼渲染 你根本不需要N卡配贵死的眼镜了，只要随便有某家的眼镜就可以搞，标准API，这边说VR/AR是潮流 那边又说UWP没用是属于因为缺少相关线索导致的精神分裂。

4. 真跨平台 就知道说可能手机和pc一起跑， 先不说小IoT板子现在跑得多欢实，就说 Hololens 我作为中国做过实机开发培训的二人之一可以告诉大家，UWP是在Hololens上直接跑成一个3d的窗口，直接拉窗体钉在墙上，是窗，是真窗，是真视窗，
统一操作方式 统一响应式ui直接看你心情来调整窗户就好了。

5 对桌面来说 将来UWP并不一定意味着可以到处都跑，而可能只是一种升级的包发布方式，这里就要说Project C了,目前虽然还没有Release，但是目前我知道的信息来看不会砍哦，他可以把一个传统Win32程序打包成一个UWP.运行起来还是你的win32 程序，但是它活在沙箱里，不能改你的主机总注册表、文件系统，带来的效果就是卸载干净，购买方便，全家桶滚蛋，如果你是个老外，你装了UWP打包过的MHO，就不用火烧火燎的问这个可不可以删除 直接卸载MHO就都没了。ok这话跟OSX的用户说不着，因为他们已经这样了，我们只能说我们网速比苹果快了。 所以说UWP并没有说一定要跨平台，也没有说一定要用XX来写App.

------针对题目来回答-------

> Windows 10 通用应用的前景如何? 修改
> Win10 Mobile 想借桌面的霸主地位，利用通用架构改善生态质量，大家感觉前途怎么样?

你问的Win10M手机 还是Win10M将来适配的所有设备 还是 UWP App?

如果你问UWP会不会有前途，我跟你说这是真正的钦点，Win10在 UWP根本就不可能没前途 就算win10失败了，也不会有人维护win7了，难道还能卸载掉设置、开始和通知中心吗 难道win10+x 就会放弃UWP吗？ 2y2sp

如果你问UWP开发者有没有前途，我觉得假如开发得烂， 干什么都没有前途。
假如你开发得好，你就算开发的是天天过马路，你都瓢满钵满。

你问Win10M的前途，这要看下面别的部门怎么搞 和我UWP关系可能相关，但是目前我们UWP只能提供更低廉的开发成本和更强大的未来科技API，并不是决定性的因素。

发布于 2016-01-17

▲ 114 ▼ 56 条评论 分享 收藏 感谢 ··· 收起

Belleve
编程 话题的优秀回答者

25 人赞同了该回答

核心问题是 API 好么，「传统的」win32 C++ 有多少坑跳过的都知道（比如著名的 A 和 W、著名的 MAX_PATH。手动斜 @vczh）； .NET （和 Java）冷启动时间和部署至今无解（0.1s 出现窗口这是硬要求）；结果现在写 win32 开发最简单的成 Electron 了……UWP 至少依 Office 里的人的说法是填坑的大好机会。PC 端的改革才是重中之重的好吗？！ @vczh 说他们正在全功能移植 Office，文档显示一个像素都不差的那种。
你们为啥都盯着手机……
交互改起来很容易（看向 android），但是 API 要重做就难了。

编辑于 2016-01-17

▲ 25 ▼ 31 条评论 分享 收藏 感谢 ···

CoderAfterWork
公众号：程序员的下班生活

7 人赞同了该回答

补充一个思路。

UWP 不只是为了在pc,平板和手机上。

微软还考虑IOT设备，以及增强现实设备的。

目标是所有设备都用一套可以代码有效重用的UWP。

编辑于 2016-01-17

▲ 7 ▼ 添加评论 分享 收藏 感谢 ···

Lucifer
米粉, 河粉, 凉粉

10 人赞同了该回答

uwp的最大敌人不是别人，正是自己的win7。
如果是在国内，还要加上xp。

发布于 2016-01-16

▲ 10 ▼ 添加评论 分享 收藏 感谢 ···

▲ 361 38 条评论