Cygwin系列（十）：折腾终端1

silaoA
爱好编程的电气工程师，silaoa.github.io
已关注

发布于 2019-12-30 17:24

> 本文共5200余字，预计阅读时间16分钟，本文知乎连接：Cygwin系列（十）：折腾终端1，本文同步发布于silaoA的博客和微信公众号平台。
> 关注学习了解更多的Cygwin、Linux、Python技术。

目 录

关于终端的概念，已在Linux Cygwin知识库（一）：一文搞清控制台、终端、shell概念中论述。UNIX/Linux系统对命令行有着天然的支持，Windows对命令行生态却不怎么重视，本着死磕自己娱乐大家的精神，深度扒一扒命令行生态中的重要元素——终端。

0x00 终端的历史演进

- **终端设备**

UNIX上古时期，经历了电传打字机（Teletype, tty）、电子视频终端（Video Terminal，键盘+阴极射线管显示器）用作终端设备（Terminal），通过RS232串口线连接主机，是计算机系统中的io外设。

在没有图形界面、只有文本字符的年代，为了更加灵活、友好地和用户交互，终端设备逐渐支持颜色、高亮、光标移动、清屏等多种效果，设备厂商定义一些特殊字符来表征这些效果，因为不是真正的输出文本内容，也被称为**转义字符**或**控制字符**，比如在shell提示符变量 `PS1` 、 `ls` 输出颜色控制变量 `LS_COLOR` 中，经常看到的"\033[34m"或"\e[34m"。这些特殊功能以函数库（termcap/terminfo）的形式提供给开发者，方便开发者调用。但是如果每个厂家都来这么一套函数库，而且各家规范也不一致的话，开发者用起来就比较头疼，程序在不同终端上兼容性也差。美国国家标准学会(American National Standard Institute，ANSI)牵头制订了包括ASCII在内的一系列规范，解决计算机、终端设备等之间数字信息问题，ANSI采用了ANSI X3.64规范，提出了“ANSI Escape Sequence”的概念（以下称“ANSI转义序列”），见维基百科ANSI_escape_code。

1978年，Digital VT100成为首个支持ANSI Escape Sequence的终端设备，随着VT100大获成功，越来越多的终端设备兼容VT100，**VT100成为了事实标准**，相关标准被UNIX、Linux所支持和继承，甚至阴极射线管显示器支持80行×24列字符的尺寸标准也被保留。

- **伪终端**

随着计算机造价变低、PC普及，终端设备退出历史舞台，特别是图形界面（GUI）技术广泛应用后，终端模拟器（Terminal Emulator）开始替代它的位置，以下不刻意区分术语“终端”和“终端模拟器”。但是命令行程序还是期望处于真实的终端设备环境中，总不能把以前所有的命令行程序中的io功能、ANSI转义序列等全部改掉，UNIX/Linux内核发展出了**伪终端**（pseudo tty，缩写为pty）设备顺应这一趋势。

UNIX/Linux系统上的终端模拟器就顺着pty这条线发展，比如X Window System的标配程序 `xterm` ，其他诸如 `rxvt` 、Gnome图形环境标配的 `Gnome Terminal` 、KDE图形环境标配的 `Konsole` 等，基本都是在 `xterm` 基础上发展而来。Cygwin通过模拟UNIX，也兼容了pty（也许只能部分兼容，未实考）。

经常可以看到终端模拟器的配置中，涉及“终端模式”或者“兼容模式”，可设置的选项包括“VT100”、“ANSI”、“xterm”等。具体属性信息，可通过 `stty` 命令显示。

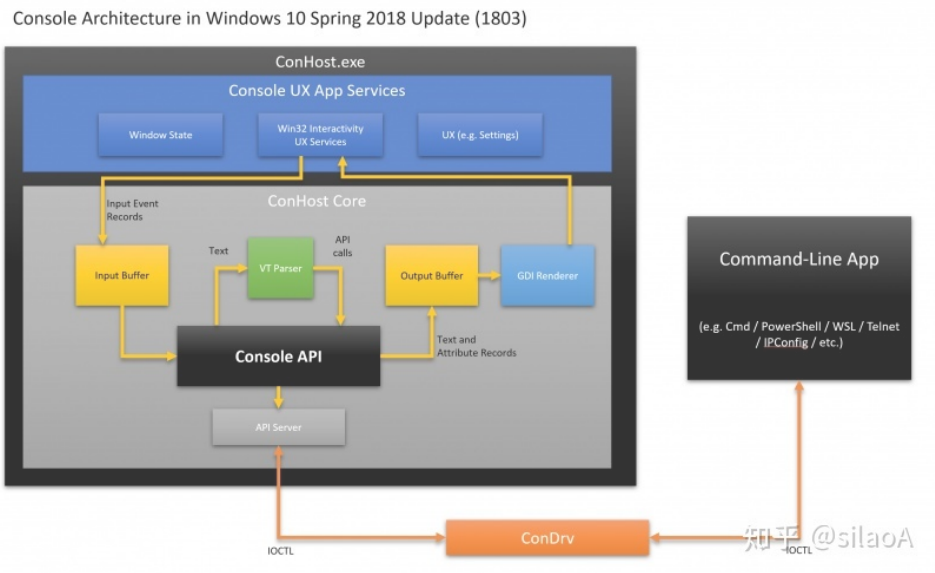
0x01 Windows对终端的支持

Windows自win95起就已成为成熟的、基于GUI的操作系统，但也未抛弃命令行程序。Windows将应用程序分为**Console应用程序**（也就是命令行程序）和**GUI应用程序**，两类应用程序所允许调用的Windows API范围不同，如果用过微软的Visual C++、Visual Studio开发工具，想必还记得在新建工程时可选工程类型就包括这两种。

由于设计理念不同，Windows内核不支持Single Unix Specification中规定的终端、伪终端设备，也不支持ANSI转义序列，但提供了Windows Console组件和相应配套的API。

- **Windows Console组件**

利用Windows Console组件，可以给命令行程序提供一个（模拟的）终端环境。从Windows 7开始，出于安全考虑，Windows Console组件整合为由以下几个二进制文件组成： - `conhost.exe` ，是一个标准的Win32 GUI应用程序，在 `C:\WINDOWS\system32` 路径下，提供一个模拟终端的窗口，负责维护存储键盘、鼠标、触屏板事件的输入队列，和存储命令行输出文本及显示属性的输出队列，支持ANSI转义序列解析、处理，管理Console窗口的布局、尺寸、位置等； - `condrv.sys` ，是内核态驱动，为 `conhost.exe` 和命令行程序提供通信通道，传递IOCTL消息。



Win10 1803上的Console架构 图片来源：微软博客

Win10 1803上的Console架构 图片来源：微软博客

Windows Console与UNIX/Linux上的pty、终端模拟器，初看起来像，但本质截然不同。

1. **Windows Console与命令行程序之间，传递的是API调用和IOCTL消息数据**，即使是嵌在文本中的ANSI转义序列也被转换为等效的Windows API调用，而UNIX/Linux上的pty设备/终端模拟器与命令行程序专递的就是文本流。这是两者设计思想的差异：在UNIX/Linux中，一切皆文件；而在Windows中，一切皆对象。

2. **Windows提供了pty设备、终端模拟器职责之外的功能特性，用于操作终端窗口**，比如Windows选择不信任命令行程序的开发者，为避免命令行程序生成ANSI转义序列，提供等效的Console API调用，操控模拟终端窗口的外观；再比如，为避免每个命令行shell重复实现命令历史、命令匿名等基础功能，Console API就实现了这些功能。

3. 最操蛋的是，**Windows强制规定命令行程序启动后，只能连接到 `conhost` 实例**，当命令行程序没有父进程或父进程是GUI程序时（比如鼠标双击文件名启动），Winodws新建一个 `conhost` 实例，强制通过 `condrv` 与之连接；当命令行程序父进程也是命令行程序时（比如在终端窗口中输入文件名启动），就继续使用父进程已连接的 `conhost` 实例与之连接。

- **Windows Console的不足**

由于上述第1点和第2点，Windows Console没有必要支持ANSI转义序列，事实上直到Windows 10，Windows Console才开始支持ANSI转义序列直接解析，在此之前Windows Console无法和UNIX/Linux上的命令行程序传递能被双方都接受的数据，导致Windows Console局限在Windows平台，UNIX/Linux上的终端模拟器也不能与远程的Windows命令行程序连接。

由于上述第3点，双击 `C:\WINDOWS\system32\cmd.exe` 或在开始菜单点击命令提示符、PowerShell等，Windows会新建一个 `conhost` 实例，并通过 `condrv` 建立通信通道，外观表现就是新建了一个模拟终端的窗口，打印着 `cmd` 或 `PowerShell` 的提示符，等待用户输入，容易被用户误认为 `cmd` 或命令提示符或 `PowerShell` 就是终端本身。在此纠正，**他们都仅仅是解释器，是常规的Windows命令行程序，扮演的是shell角色，与UNIX/Linux上的 `bash` 、 `zsh` 对等，也不是什么dos程序**，而终端模拟器是Win32 GUI程序。此外，双击 `C:\WINDOWS\system32\conhost.exe` 文件名，或在终端窗口中输入"conhost"，Windows会新建一个 `conhost` 实例，并且默认启动一个解释器，默认值通常是 `cmd` ，外观表现就是打开了新的模拟终端窗口，里面打印着提示符，等待用户输入。

还是由于上述第3点，**Windows不给第三方终端模拟器与命令行程序通信的机会**，导致第三方终端模拟器的实现流程异常曲折。

鉴于这些不同，可以说Windows与UNIX/Linux的命令行生态，是完全互不兼容的两套。Cygwin作为UNIX兼容层，算是比较好的解决方案，尽可能抹平二者差异。

0x02 第三方终端模拟器

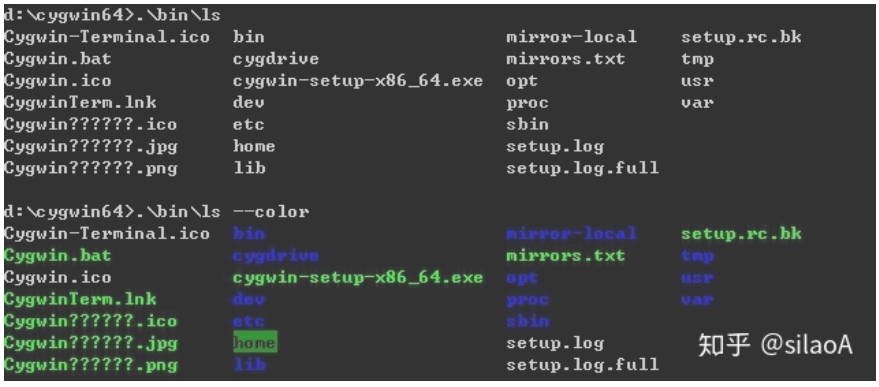
- **概览**

基于前一节内容，可将终端模拟器分为2类：基于Windows Console组件的和基于pty的。

鉴于前一节所述第3点，基于Windows Console的第三方终端实现流程通常是：

- 启动 `conhost` 进程并将 `conhost` 窗口隐藏起来，比如将窗口移到负数坐标位置；
- 绘制自己的一套UI，记录按键、鼠标、触屏事件，同时发送给 `conhost` 进程，最终传递给命令行程序，命令行程序照常执行；
- 抓取隐藏的 `conhost` 窗口上的命令行程序输出的文本内容、风格，再在自己的UI上渲染出来。以上流程，**看起来就像是个非法软件**采集用户输入，同时也把输出呈现在屏幕上，让用户误以为是第三方终端模拟器连接了命令行程序，但在任务管理器中查看，必然可见conhost.exe。与此同时，如何直接解析ANSI转义序列也需要第三方终端模拟器自己想办法，或者抱Cygwin大腿。

基于pty的，通常都是抱Cygwin大腿，捎带脚把ANSI转义序列解析也一起抱上了，但还有个情况需要说明。Cygwin核心以及有些命令行程序，能探测到输出定向至哪里，比如是管道（pipe在UNIX/Linux中十分常见），还是基于pty的终端，或是基于Windows Console的终端。当探测到是基于pty的终端时，Cygwin核心把ANSI转义序列原封不动地传递给终端，完全交由终端自行渲染；而探测到是Windows Console终端的时候，就自己先处理一下再交由终端渲染。也就是说，**对于基于Windows Console的终端，Cygwin并不清楚它有多强的功能，于是选择了不信任，自己上**！这样**导致即使有的第三方终端模拟器有处理ANSI转义序列的能力，也无法很好地运行Cygwin命令行程序**，下文会介绍。



Windows7 conhost中运行cygwin的ls

Windows7 mintty中运行cygwin的ls

如上两图对比效果。注意，本文的Cygwin Shell环境中设置了匿名 `alias ls=ls --color` ，所以不必加 `--color` 选项，而 `cmd` 解释器是不知道Cygwin Shell配置情况的，在conhost窗口中输入命令要加上 `--color` 选项才显示颜色。另外，由于字符集不统一，conhost窗口把 `ls` 命令输出的中文显示成“?”。

尽管困难重重，Windows上还是诞生了许多优秀的第三方终端模拟器，有的还把ssh/串口/ftp客户端程序，甚至其他常用命令行程序等也一起打包发行，已经不是单纯的终端模拟器了。以下仅列举几个。

- **Mintty**

Mintty是一款基于pty的开源终端，从 `putty` 0.60版本的代码分支而来，现今项目地址：https://mintty.github.io，兼容POSIX “pipe”模式，完全支持ANSI转义序列。不同寻常的是，在抱了Cygwin大腿的同时，还调用了Win32 GUI API，运行不仅依赖Cygwin核心（ `cygwin1.dll` ），还依赖一堆Windows的dll（ `msvcrt.dll` 、 `GDI32.dll` 、 `USER32.dll` 、

KERNEL32.DLL ）等。它不仅是Cygwin的默认终端，也是MSYS、MSYS2等的默认终端，随各项目的软件包发行，与 wslbrige 配合还能连接到WSL（的命令行程序）。从这可以看出，Mintty和Cygwin关系密切不一般，官方默认没毛病。

Mintty的主要功能特性：

- 兼容 xterm ，支持DEC公司（VT100就是他家的）终端产品所有屏幕控制功能；
- 支持字符加粗、斜体、下划线、闪烁、前后景色等风格；
- 全面支持Unicode，支持Emoji；
- 支持256色和真彩色；
- 支持图片显示；
- 图形化配置，保存为配置文件 .mintty ；
- 灵活的文字复制粘贴操作；
- 支持Windows XP至Windows10。

Mintty项目官网列出展示上述功能特性的截图，还提供了使用手册，使用手册也是随软件包一起发行。Cygwin系列（五）：Shell命令行初体验也简单说了它的工作原理。

Mintty的强大之处无需赘述，但不可避免地存在一个问题——不能很好地和Windows命令行程序配合起来，一方面前文陈述的Windows Console组件和pty的不兼容，另一方面是Windows命令行程序使用的字符集与Mintty所支持的存在不兼容。还有我个人觉得不太爽的，**尽管支持通过如 tmux 、 GNU Screen 等多路复用器切换窗口，但Mintty不支持多标签页。**

- **ConEmu**

ConEmu是一款优秀的开源终端，项目地址https://conemu.github.io。它是混合型的，首要支持Windows Console功能，可看作是 conhost 的增强版本，同时也部分地支持pty，具备ANSI转义序列直接解析能力，总体上是 conhost + Mintty 的全能选手。

主要功能特性包括：

- **多标签页、多标签页、多标签页**，重要事情说3遍，并且支持多种布局方式；
- **不同标签页独立配置Task**，也就是运行不同的命令行程序，特别是shell，Task支持名字分组和分配快捷键；
- 界面现代化，有标签页、快捷菜单、控制台区、状态栏、滚动条；
- 支持连接到多种类型命令行程序环境，Cygwin、msys、WSL都不在话下；
- 图形化配置方式，每一部分都支持个性化配置，所有配置项保存到文件 ConEmu.xml ；
- 支持集成到其他程序界面中，比如文件管理器Explorer；
- 支持字符加粗、斜体、下划线、闪烁、前后景色等风格；
- 全面支持Unicode，支持ANSI X3.64标准和 xterm 256色；
- 附带字符界面的文本管理器Far Manager及其插件；
- 能运行在Windows 2000及以上版本的Windows系统，开箱即用。

由于前文所述原因，ConEmu与Cygwin命令行程序的配合暂时还不完美，等未来ConEmu完全支持pty，能骗过Cygwin了，ANSI转义序列直接解析能力才有用武之地。为此，ConEmu临时提供了间接解决办法——cygwin terminal connector。随ConEmu打包的 conemu-cyg-64.exe 或 conemu-cyg-32.exe 程序就是一个terminal connector，本身也是个Cygwin程序，依赖 cygwin1.dll 。ConEmu通过它启动Cygwin命令行程序，并强迫Cygwin核心关掉ANSI转义序列处理，原封不动地交给ConEmu，这好比在Cygwin中安插间谍，抓取第一手情报，但仍只是部分地解决了问题，比如颜色显示就存在不一致。

ConEmu官方提供了一个在ConEmu内连接Cygwin bash的Task配置示例。

```
set "PATH=C:\cygwin64\bin;%PATH%" &
%ConEmuBaseDirShort%\conemu-cyg-64.exe /usr/bin/bash.exe --login -i
-new_console:p:C:"C:\cygwin64\Cygwin.ico"
```

解释如下：

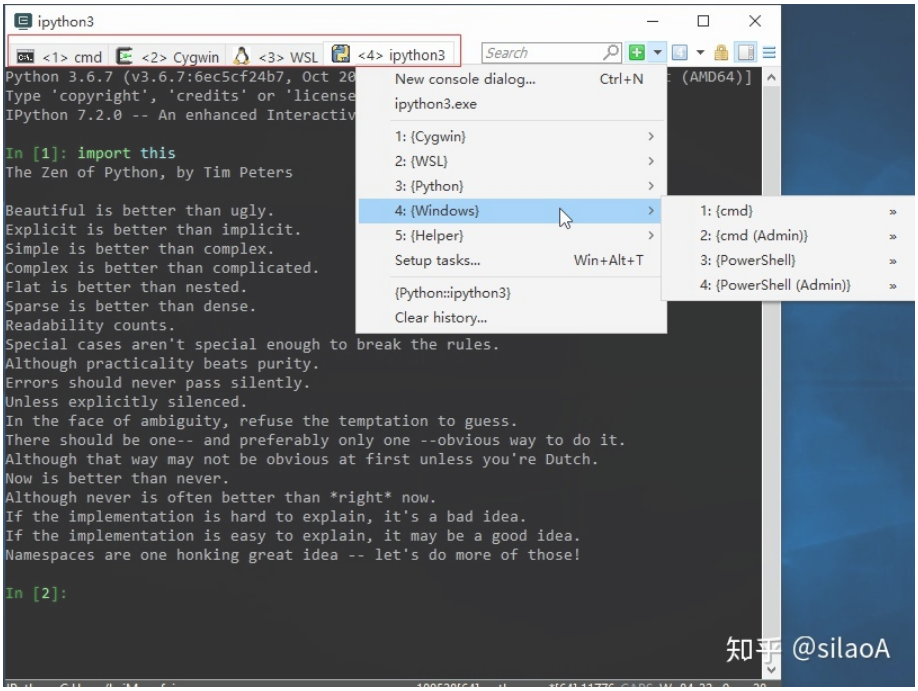
- 将 C:\cygwin64\bin 加到Windows PATH环境变量是为了让 conemu-cyg-64.exe 准确找到

Cygwin核心（ `cygwin1.dll` ），明确Cygwin根路径；
- `/usr/bin/bash.exe --login -i` 是指定ConEmu要连接的命令行程序及其参数，此处是连接到Cygwin的bash，也可以配置为其他；
- 最后一行是ConEmu特有的选项开关，选项间用":"隔开， `-new_console` 含义是打开新标签页， `p` 是指定pty模式， `C:\cygwin64\Cygwin.ico` 是指定标签页使用的图标文件。

更详细解释及其他有效配置见官方文档 conemu.github.io/en/Cyg...。ConEmu还打包了msys、msys2的terminal connector—— `conemu-msys-32.exe` 和 `conemu-msys2-32/64.exe` ；给WSL的terminal connector和Cygwin的一样，也是需要与 `wslbrige` 配合起来，官网也提供了连接到msys、msys2、WSL的Task配置示例。搭配一个Windows的ssh客户端程序，配置好Task，也能将ConEmu连接到远程的ssh server。

配置Task较多，分组的作用就体现出来了。比如把Windows中的cmd、Power Shell、python shell、ipython都分到Windows组，在Cygwin中有bash、zsh、fish、ipython等等，可以归并到Cygwin组，如果同时还用msys、WSL什么的，也可以相应增加分组。这样可以做到**每种环境独立划分一个组**，不至于同名的程序难以区分，界面上也干净清爽。

如图有多个分组，已同时打开4个不同标签页，可用ctrl+TAB键顺序切换或ctrl`数字键任意跳转。



ConEmu界面

最后，**ConEmu真香**！

其他

- cmder是一款默认配置十分漂亮的开源终端，项目地址：https://cmder.net。cmder是基于ConEmu而开发的，它分为mini版和full版，mini版基本就是一套ConEmu了，full版就是把git bash for Windows也打包在一起。
- console2、consoleZ都是基于Windows Console的开源终端，支持多标签页，是 `conhost` 的包装。console2项目地址 sourceforge.net/project...，由于长久未更新，才有了分支consoleZ，后者项目地址github.com/cbucher/cons...，支持连接到cmd/Power Shell、Cygwin、msys(2)，**界面与Mintty极其相似。**
- git bash for Windows是git官方给Windows平台做的git打包，可以看作是msys(2)的子集，使用的终端是Mintty，另外把整套git和少数UNIX常用命令行程序打包在一起。
- Xshell、MobaXterm均为商业产品，界面华丽，不是单纯的终端了，它们把telnet、ssh、ftp这些常用客户端程序也打包进来了，MobaXterm甚至把x server都打包进来。

0x03 总结

历史发展造成了UNIX与Windows平台互不兼容的命令行生态，UNIX/Linux的开放性使得第三方终端繁荣发展，Windows的封闭性和对命令行的忽视造给第三方终端增加重重困难，尽管如此还是诞生了优秀的终端模拟器。经过使用，本人强烈安利ConEmu替代上述所有。

填补Windows Console和pty的差异性，不只是Cygwin这一条"重量级"解决方案，敬请期待下一篇继续深度扒终端。

参考

- Single Unix Specification.
- 维基百科：ANSI_escape_code.
- 维基百科：Win32 控制台.
- Windows Command-Line: Inside the Windows Console.
- https://mintty.github.io
- https://conemu.github.io
- github.com/cbucher/cons...

更多阅读

- Linux Cygwin知识库（一）：一文搞清控制台、终端、shell概念
- Cygwin系列（五）：Shell命令行初体验
- Cygwin系列（九）：Cygwin学习路线
- Python操作Excel文件（0）：盘点
- 微软WSL——Linux桌面版未来之光
- 专栏：伪码人We_Coder
- GNU Wget 爬虫？试一试
- silaoA的博客.https://silaoa.github.io

如本文对你有帮助，或内容引起极度舒适，欢迎分享转发与留言交流

weixin.qq.com/r/rR2VjSP... (二维码自动识别)

qm.qq.com/cgi-bin/qm/qr?... (二维码自动识别)

最后，好书推荐

发布于 2019-12-30 17:24

Linux Cygwin Microsoft Windows

写评论... | 你赞过作者的回答

还没有评论，发表第一个评论吧

文章被以下专栏收录

伪码人 We_Coder
专注Cygwin、Linux、电气等技术交流分享

我爱命令行
命令行使用经验分享

Linux 漫游之旅
还有什么能阻挡我学习 Linux?