天上人间

MSYS2详解

前言

很多人都觉得 Linux 相比于 Windows 而言更适合开发，但由于 Windows 在 PC 上占有量巨大，一些程序即使原本在 Linux 上开发的，最终往往需要部署到 Windows 环境。面对这样的跨平台需求，比较常见的操作是用宏来控制程序的编译：

```
#if defined(_MSC_VER) || defined(_WIN32) || defined(_WIN64)
#define DLL_EXPORT __declspec(dllexport)
#else
#define DLL_EXPORT
#endif

DLL_EXPORT void func();
```

利用宏可以在不同的平台上生成不同的代码，从而让一个工程同时支持 Linux (gcc) 和 Windows (MSVC)。不过这种方法增加了编写程序时需要考虑的因素（系统相关或者编译器相关的代码需要特殊处理，甚至需要写两种实现），而且如果程序原本只考虑了 gcc，后面想移植 MSVC 可能还是要费点脑筋的。另外，MSVC 有时会存在一些莫名其妙的 bug，比如：

相比而言，gcc 貌似就没碰上过 bug（也有可能时因为 Manjaro 滚动更新，bug 修的快）。

考虑到 MSVC 的额外工作量，以及对 gcc 的偏好，C++ 程序移植 Windows 其实还有另一种办法：`MinGW`

MinGW 是什么

MinGW (Minimalist GNU for Windows) 是一种用于开发原生 Windows 应用的最小化 GNU 开发环境，可以理解为开发 Windows 程序的 gcc，也就是说这个gcc就是专门编译出能在windows平台运行的程序的编译器。MinGW 本身并不一定要运行在 Windows 下，Linux 上也可以通过 MinGW 工具链交叉编译 Windows 程序。

MSYS (Minimal SYStem) 是一系列 GNU 工具（bash、make、gawk、grep等）的集合（基于旧版 Cygwin），用于弥补 Windows cmd shell 的不足，让 MinGW 在 Windows 上更便于使用。

MSYS2 是一个独立的 MSYS 重写（因为 MinGW 和 MSYS 更新缓慢），基于新一代 Cygwin 和 MinGW-w64，提供更多 API 支持和 64 位应用开发，因此建议抛弃 MSYS 直接使用 MSYS2。

至于 Cygwin 是什么以及其它详细介绍，可参见相关官网和 Cygwin 和MinGW 的区别与联系是怎样的？@LiTuX 的回答。

一、MSYS2 介绍

MSYS2（Minimal SYStem 2）是一个集成了大量的GNU工具链、工具和库的开源软件包集合。**它提供了一个类似于Linux的shell环境，可以在Windows系统中编译和运行许多Linux应用程序和工具。**

MSYS2基于MinGW-w64平台，提供了一个完整的开发环境，包括GCC编译器、GDB调试器、Make、Git版本控制系统和许多其他开发工具。除了常用的开发库和工具之外，MSYS2还提供了许多专门针对Windows平台的库和工具，方便开发人员进行跨平台开发和移植工作。

由于MSYS2拥有比较完整的Linux工具链和库，因此它成为了许多跨平台开发和移植项目的首选工具。另外，使用MSYS2也可以轻松地在Windows系统中搭建一个类似于Linux的软件开发环境，方便开发人员进行开发和调试工作。

二、MSYS2、Cygwin、MinGW-w64介绍

`MSYS2`是一个包含MinGW-w64工具链、GNU工具集和一些开源库的平台，它提供了一种在Windows上编译和运行Unix/Linux程序的方式。MSYS2与MinGW-w64相似，但比MinGW-w64更完整和稳定，提供了Pacman包管理器以方便用户安装和管理软件包。

`MinGW-w64`是一个Windows下的C/C++编程工具集，它提供了运行在Windows上的GNU工具集和GCC编译器。MinGW-w64与MSYS2类似，但主要用于编译Windows本地应用程序，而非Unix/Linux程序。MinGW-w64也可以用于交叉编译，为其他平台生成Windows可执行文件。

区别：

MSYS： 相当于操作系统（如Windows），这个操作系统提供的软件、接口等和Linux相似。

MinGW： 相当于开发工具包（如MSVC），这个开发工具包可以运行在 MSYS 下，包里的工具也可以运行在Windows下，编译结果是Windows程序。

`Cygwin`是一个在Windows平台上运行的兼容性层，提供了类Unix环境的工具和开发库。Cygwin将Unix程序编译为Windows本地代码，然后在Windows上运行。它提供了最完整的Linux/Unix环境，但相对于MSYS2和MinGW-w64，Cygwin的性能较差。

因此，MSYS2适用于需要在Windows上编译和运行Unix/Linux程序的场景，MinGW-w64适用于编译Windows本地应用程序的场景，Cygwin适用于需要最完整的Linux/Unix环境的场景。

msys2的优势： 简单的包管理工具，不需网上搜索安装包，下载安装。直接运用pacman进行下载，并且升级简单。

三、下载安装

MSYS2官网：https://www.msys2.org/

MSYS2 Packages查询网站：https://packages.msys2.org/queue

安装包下载

- github：https://github.com/msys2/msys2-installer/releases
- 官方仓库：https://repo.msys2.org/distrib/

安装包有三种：

- `msys2-x86_64-*.exe`: GUI安装程序(windows推荐这种)
- `msys2-base-x86_64-*.sfx.exe`: 只是自解压归档中的文件(缺少Windows集成，如 快捷方式，卸载入口，但其它工作相同)
- `msys2-base-x86_64-*. tar.xz`: 与.sfx.exe相同，但作为xz归档文件

0、msys2本质上是一个裁剪版的操作系统平台，在这个平台下可以安装各种gcc工具链，它可以管理MGW、Clang等编译工具链。所以使用msys2安装基于msys2项目提供的MGW64，就不用单独安装基于MGW64项目提供的MWG64编译器了

1、安装路径要求：要求卷类型为NTFS，路径只有ascii字符，没有重音，没有空格，没有符号链接，没有subst或网络驱动器，不能是FAT

2、msys2安装会一直卡在50%，会无法安装成功(断网安装就可以成功：https://tieba.baidu.com/p/6193884364)

3、修改包下载的镜像源,提升依赖的下载速度(可以FQ的话就不用配置这个源)
修改 `D:\msys64\etc\pacman.d` 目录下的所有 `mirrorlist.*` 文件，将原来的注释掉，添加如下

```
#清华源
Server = http://mirrors.tuna.tsinghua.edu.cn
#中科大源
Server = http://mirrors.ustc.edu.cn
#北理源
Server = http://mirror.bit.edu.cn
```

4、更新依赖

第一次执行更新 MSYS2 核心包：`pacman -Suy`

第二次执行更新 MSYS2 非核心包：`pacman -Suy`

5、安装64位windows10编译工具链

- pacman -S mingw-w64-x86_64-gcc
- pacman -S mingw-w64-x86_64-cmake-gui
- pacman -S mingw-w64-x86_64-gdb
- pacman -S mingw-w64-x86_64-make
- pacman -S mingw-w64-x86_64-boost
- pacman -S mingw-w64-x86_64-opencv

6、设置环境变量
添加目录 `;D:\msys64\mingw64\bin;D:\msys64;` 到path中

四、pacman介绍

0、查询已安装的软件包(简单信息)：`pacman -Q`

1、查询已安装的软件包(详细信息)：`pacman -Qs`

2、查询从 Arch Linux 官方软件仓库安装的软件包：`pacman -Qe`

3、查询某个软件包详细信息：`pacman -Qi 完整包名`

4、搜索软件包：`pacman -Ss 关键字`

5、安装软件包：`pacman -S 包名`

6、下载软件包但不安装：`pacman -Sw 包名`

7、从归档包安装：`pacman -U [包名.tar.zst | 包名.tar.xz]`
归档包仓库：https://repo.msys2.org/

8、卸载软件包：`pacman -R 包名`
注意，该命令仅卸载软件包，无法卸载该软件包安装时的依赖包，也无法移除该软件运行过程中产生的任何文件。

9、列出所有的软件包组：`pacman -Sg`

10、列出某个软件包组中所有的软件包：`pacman -Sg 组名`

11、查 pacman 用法

```
$ pacman -h
usage:  pacman <operation> [...]
operations:
    pacman {-h --help}
    pacman {-V --version}
    pacman {-D --database} <options> <package(s)>
    pacman {-F --files}    [options] [file(s)]
    pacman {-Q --query}    [options] [package(s)]
    pacman {-R --remove}   [options] <package(s)>
    pacman {-S --sync}     [options] [package(s)]
    pacman {-T --deptest}  [options] [package(s)]
    pacman {-U --upgrade}  [options] <file(s)>
```

12、查 pacman 选项详细用法

```
$ pacman -S -h
usage:  pacman {-S --sync} [options] [package(s)]
options:
  -b, --dbpath <path>  set an alternate database location
  -c, --clean          remove old packages from cache directory (-cc for all)
  -d, --nodeps         skip dependency version checks (-dd to skip all checks)
  -g, --groups         view all members of a package group
                       (-gg to view all groups and members)
  -i, --info           view package information (-ii for extended information)
  -l, --list <repo>    view a list of packages in a repo
  -p, --print          print the targets instead of performing the operation
  -q, --quiet          show less information for query and search
  -r, --root <path>    set an alternate installation root
  -s, --search <regex> search remote repositories for matching strings
  -u, --sysupgrade     upgrade installed packages (-uu enables downgrades)
  -v, --verbose        be verbose
  -w, --downloadonly   download packages but do not install/upgrade anything
  -y, --refresh        download fresh package databases from the server
                       (-yy to force a refresh even if up to date)
      --arch <arch>    set an alternate architecture
      --asdeps         install packages as non-explicitly installed
      --asexplicit     install packages as explicitly installed
      --assume-installed <package=version>
                       add a virtual package to satisfy dependencies
      --cachedir <dir> set an alternate package cache location
      --color <when>   colorize the output
      --config <path>  set an alternate configuration file
      --confirm        always ask for confirmation
      --dbonly         only modify database entries, not package files
      --debug          display debug messages
      --disable-download-timeout
                       use relaxed timeouts for download
      --gpgdir <path>  set an alternate home directory for GnuPG
      --hookdir <dir>  set an alternate hook location
      --ignore <pkg>   ignore a package upgrade (can be used more than once)
      --ignoregroup <grp>
                       ignore a group upgrade (can be used more than once)
      --logfile <path> set an alternate log file
      --needed         do not reinstall up to date packages
      --noconfirm      do not ask for any confirmation
      --noprogressbar  do not show a progress bar when downloading files
      --noscriptlet    do not execute the install scriptlet if one exists
      --overwrite <glob>
                       overwrite conflicting files (can be used more than once)
      --print-format <string>
                       specify how the targets should be printed
      --sysroot        operate on a mounted guest system (root-only)
```

参考

MSYS2 介绍、下载与安装、Pacman常用命令：
https://blog.csdn.net/B11050729/article/details/131719012

使用MinGW(MSYS2)在Windows上部署Linux C++程序：
https://blog.csdn.net/mightbxg/article/details/108484824
分类: C++相关
标签: C++