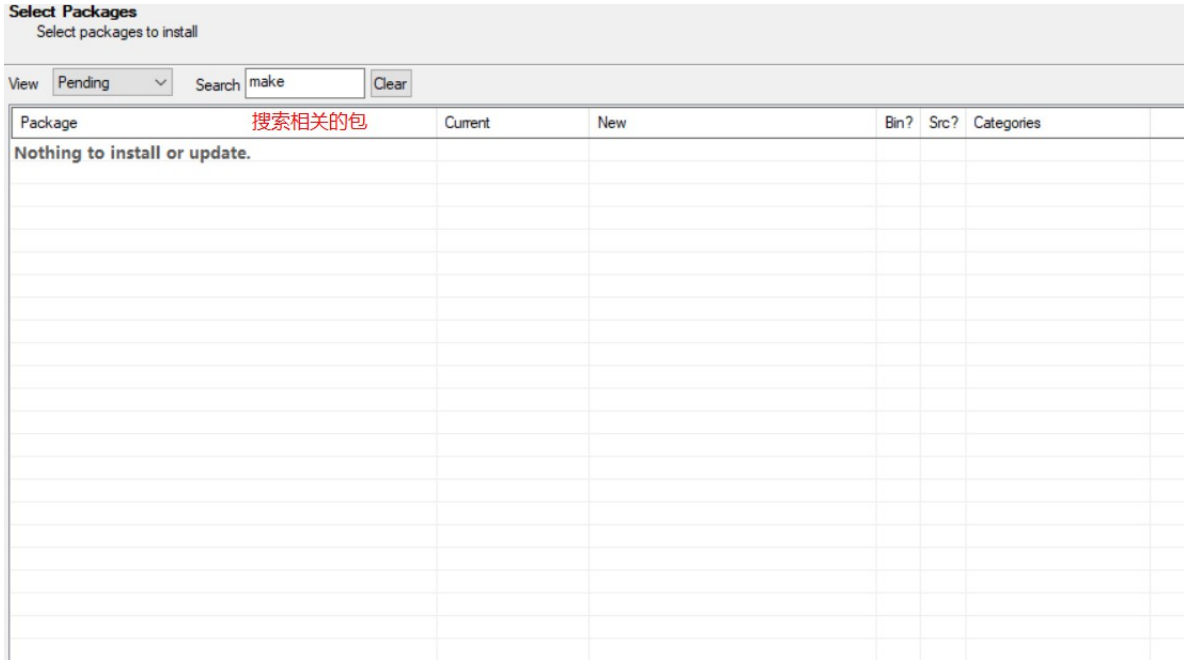
VST 在 Win 10 下安装（利用 Cygwin）

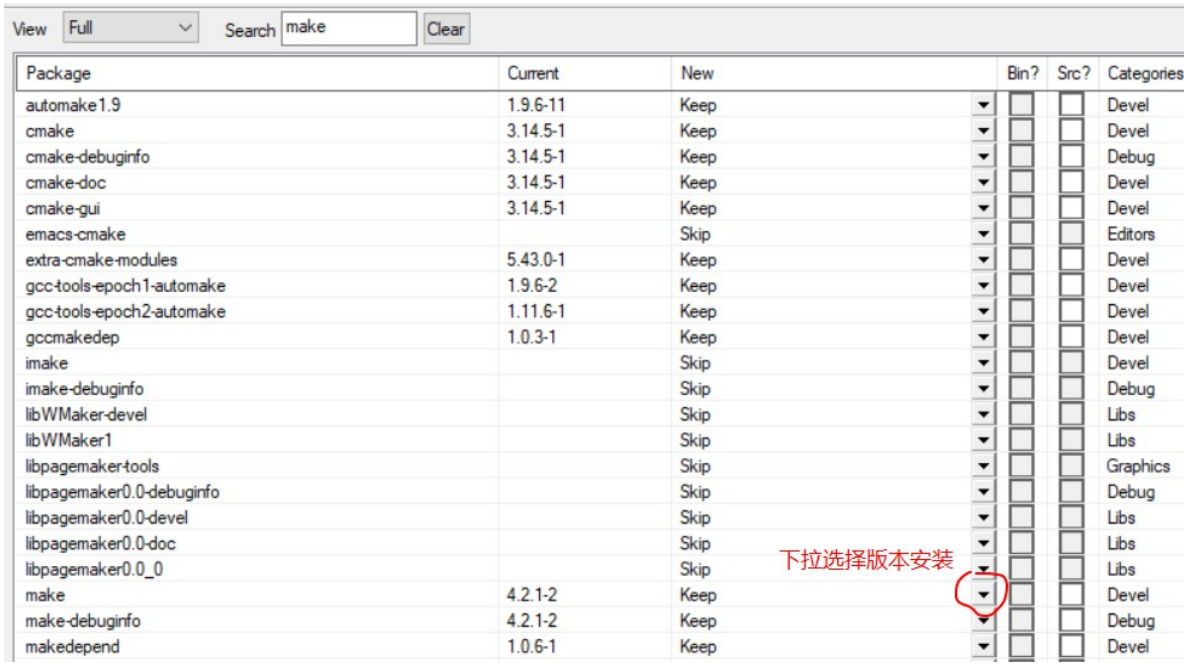
准备工作

先下载 VST 的安装包：https://github.com/PrincetonUniversity/VST（或者直接 git clone），解压到某个目录备用。

然后下载对应版本（64 位或 32 位）的 Cygwin：http://www.cygwin.com/

安装 Cygwin，进入包安装界面（如下图），尝试安装下面几个包：`make`，`git`，`vim`。





勾选完成这些包之后点 Next 开始安装包，务必保持网络通畅。

相关环境设置

找到 Coq 安装目录下的 bin 文件夹，保证里面有 `coqc.exe`，记录该路径（如 `D:\Coq\bin`）。

然后在 VST 的根目录下新建文件 `CONFIGURE`，在里面写入下面的内容：

```
COQBIN=/cygdrive/**你的 Coq bin 路径**
```

比如我的是 `D:\Coq\bin`，那么我就要向 `CONFIGURE` 中写入 `COQBIN=/cygdrive/D/Coq/bin/`。写完最好换行。

编译

打开 Cygwin 的终端。

定位到 VST 的根目录下。注意：要先执行 `cd /cygdrive` 进入 Cygwin 的虚拟目录，然后才能像在 CMD 中一样进行定位操作。这里以定位到 `D:\VST\` 为例。

然后执行 `make floyd`，开始编译。（如果你的 CPU 有多个核心，可以执行 `make floyd -j3` 以加快速度）

编译完成后再执行 `make`，编译其他文件。

对相关库的支持

编译完成后，可以查看编译后的效果。

这里用 VST 的 Tutorial 和其使用到的 C 程序做示例。将这些文件放在 `D:\try` 中。

然后在该目录下新建 `_CoqProject` 文件。里面输入以下内容：

```
-Q . PL
-Q ../VST/msl VST.msl
-Q ../VST/sepcomp VST.sepcomp
-Q ../VST/veric VST.veric
-Q ../VST/floyd VST.floyd
-Q ../VST/progs VST.progs
-R ../VST/compcert compcert
```

后面和 VST 有关的路径需要设定为 VST 的目录。这里使用的是 `D:\try`，且 VST 的根目录为 `D:\VST` 故这么写。

然后就可以在 Coq 中尝试编译各个 .v 文件，先编译各个 C 文件对应的 .v 文件，最后编译 VST 的 Tutorial 文件。

如果我想自己写 C 验证着玩呢?

可以考虑手动安装 Compcert 然后用。但是不推荐，如果只想要用 Compcert 的将 C 程序转换为 .v 文件的功能的话只要有编译好的 clightgen.exe 就行了。

要将 C 程序转换为 .v 文件，只需要将 C 程序和 clightgen.exe 和 compcert.ini 放在同一目录下，然后用控制台执行 `clightgen -normalize xxx.c`（`xxx.c` 是你的 C 程序名）即可。

（不过并不清楚将一台机器上编译的 clightgen 放到另一台机器上有没有问题，可能是没什么问题的）

怎么证明?

从 Tutorial 就可以看出证明大概分三个部分：

1. 准备好需要验证的 C 程序，例如 `F.c`。一般我们希望证明的都是某个 C 函数的正确性，最后将这些对函数体的证明串起来，变成对 C 程序本身正确性的证明。

2. 用 clightgen 将其形式化，生成文件 `F.v`，其中包括了用 Coq 语言描述的 C 程序的 AST。将得到的 `F.v` 编译好，得到 `F.vo` 文件。
3. 写一个证明程序，记为 `verif_F.v`。`verif_F.v` 当中应该包含这些内容：
    1. 被证明程序的函数式实现。一般就是在 Coq 中用一系列语言表达被证明程序中的函数。这一部分往往还包括一些相关定理的证明。例如，我们要证明一个反转链表的函数的正确性，那么我们就可以在 Coq 中先定义列表，然后证明列表反转的一些性质，且这些性质最好是和我们程序中用到的算法相关的。
    2. 被证明程序的函数规格（Function Specification）。简单来说，这一部分就是将 Coq 中的证明和 C 中的程序连接起来的桥梁。
    3. 对函数体的证明。这一部分在 Coq 中依靠 VST 实现。

当然，这是针对比较简单的情况。复杂如多个 C 程序的场合下也基本上是分成这三个模块来做。

具体该怎么做可以参考 Verifiable C 和 VST/progs 下的各种 `verif_....v`。