<http://blog.csdn.net/solotony/article/details/45175>

BJam的简介  
-==-==-==-==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==-  
Bjam 是一个类似于Make的项目管理工具.它专门为Boost定制的编译管理器,它基于  
FTJam, FTJam是从Perforce Jam发展起来的.Bjam向后兼容Perforce Jam.

Bjam工具由Boost.Jam项目维护.

当前Bjam的最新版本是:3.1.9(与Boost-1.31.0一起发布).它基于2.4 of Jam/MR.  
其授权为:  
/+/  
+/  Copyright 1993-2002 Christopher Seiwald and Perforce Software, Inc.  
/+/  
This is Release 2.4 of Jam/MR, a make-like program.  
License is hereby granted to use this software and distribute it  
freely, as long as this copyright notice is retained and modifications  
are clearly marked.  
ALL WARRANTIES ARE HEREBY DISCLAIMED.

BJam的执行过程  
-==-==-==-==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==-

Bjam由bjam.exe和一些项目配置文件(由jam语言写成)组成.

bjam执行过程分为四个过程:

StartUp(准备)  
 bjam将所有环境变量引入jam执行环境.同时也把命令行中以-s指定的变量  
引入执行环境.

Parsing (分析)  
 bjam读入jambase文件.如果不存在,就读入内建的jambase.  
内建jambase的最后一条指令是INCLUDE(包含) jamfile 文件.这是一个由用户  
提供的文件,其作用相当于makefile.  
 jambase和jamfile的作用都是(1)定义目标和源文件,及其依赖关系  
jambase文件定义了基本的Rule(规则)和变量.jamfile使用这些Rule指明依赖  
关系.

Binding(绑定)  
 分为三个子步骤:

 Binding  
 jam循环遍历依赖树,将文件目标与真实的文件相关联.如果发现有环状依赖  
将会报错.  
 文件目标以绝对路径或相对路径表示,一般绑定到相应的文件,但通过修改  
$(SEARCH) 和 $(LOCATE)可以修改文件目标与真实文件关联的方式.见Rule参考.

 Update Determination   
 完成Binding后,jam将决定哪些文件目标需要更新.文件不存在,文件比源  
文件之一旧或者它的源文件之一标记为要更新,该目标会标记为需要更新.  
 可以通过ALWAYS, LEAVES, NOCARE, NOTFILE, NOUPDATE和TEMPORARY  
Rule修改更新规则.

 Header File Scanning   
 jam还以对文件进行了扫描,以发现C风格的头文件引用.这一行为由$(HDRSCAN)  
和$(HDRRULE)变量控制.扫描结果做为依赖关系加入依赖树中.  
   
Updating(更新)  
 完成Binding过程后,jam将循环遍历依赖树,这次将执行每个标记为需要更新  
的目标相关联的更新过程.  
   
 如果指定了-j,jam将顺序执行.

Jam语言定义  
-==-==-==-==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==-

词法  
    Jam认为所有的记号(Token)都由空白字符(blanks, tabs, or newlines)分隔.  
包括符号(:)和(;)也必须用空白字符分隔.

例外:  
    由引号(")包围的记号中, 可以使用空白符号.  
    符号(/)可以转义引号和空白字符.

    由{}包围的字符串,可以使用空白符号,并且认为只是一个字符串.

    关键字,做为记号时,必须使用引号包围.

内建Rule语义  
-==-==-==-==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==-

1.构造依赖关系

    DEPENDS targets1 : targets2 ;  
    使用target1依赖于targets2, 如果targets2比targets1新, targets1会被重  
建.  
    INCLUDES targets1 : targets2 ;  
    构造同位依赖, 使依赖于targets2的目标同时也依赖于targets1.

2.修改依赖关系

    ALWAYS targets ;

    LEAVES targets ;

    NOCARE targets ;

    NOTFILE targets ;

    NOUPDATE targets ;

    TEMPORARY targets ;

3.工具  
      
    ECHO args ;  
    显示参数

    EXIT args ;  
    显示参数,并退出.

    GLOB directories : patterns : downcase-opt  
    在directories中取出符合patterns的文件名.

    MATCH regexps : list  
    从list中取出符合regexps的符号.

流程控制  
-==-==-==-==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==-

--------------------------------  
for var in list { statements }   
--------------------------------

为list中的每个执行一次statements, var被设计为list元素的值.

--------------------------------  
if cond { statements }   
[ else statements ]   
--------------------------------

显然,else部分是可以选的.

条件可以是:  
a 当a的元素中最少有一个非空字符串时为true;  
a = b  列表元素全部相同时为true  
a != b  列表元素不完全相同时为true  
a < b  a和b中,第一个不相同的元素相比较, a的元素比b小(按字符串顺序).  
a <= b  a中每个元素都比b对应元素小或相等  
a > b  a和b中,第一个不相同的元素相比较, a的元素比b大(按字符串顺序).  
a >= b  a中每个元素都比b对应元素大或相等  
a in b  a所有元素都可以在b中找到,或者a是空表  
! cond  逻辑非   
cond && cond  逻辑与  
cond || cond  逻辑或  
( cond )  优先运算

--------------------------------  
   include file ;   
--------------------------------

使用jam读入file,并处理它.  
这个过程在解析阶段发生,所以file不会被建造,也没有文件范围.

--------------------------------  
local vars [ = values ] ;   
--------------------------------

定义一个局部变量,在{}之外原变量的值会回存.

--------------------------------  
return values ;   
--------------------------------

设置返回值,注意在rule中,return并不跳出rule的执行过程.

--------------------------------  
switch value   
{   
case pattern1 : statements ;   
case pattern2 : statements ;   
...   
}   
--------------------------------  
根据value执行一次或零次statements. pattern可以是以下符号通配符

?  匹配一个任意字符  
\*  匹配零个或更多字符  
[chars] 匹配chars中的任意一个字符   
[^chars] 匹配不在chars中的任意一个字符  
/x  匹配x(转义其它通配符)

--------------------------------  
while cond { statements }   
--------------------------------

在条件为真时,反复执行statement.

变量控制和内建变量  
-==-==-==-==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==-

Jam变量是一个字符串列表,它的元素可能是0个,也可以是多个字符串.一个未定义的变量和  
一个空变量(0个元素)没有区别.但是一个变量可以定义为含有多个空字符串.

变量可以通过$(variable)来取值

变量有"全局"或"目标专用"两种.目标专用的变量只在建造目标时取值.

变量定义的方式有以下几种.  
 variable = elements ;   
 variable += elements ;   
 variable on targets = elements ;   
 variable on targets += elements ;   
 variable default = elements ;   
 variable ?= elements ;

前两种方式定义全局变量, 第三和第四种方式定义目标专用变量.  
=号可以重写变量的内容.+=连接原有内容和新内容.  
最后两种效果一致:在变量没有定义时,定义一个全局变量.

命令行选项  
-==-==-==-==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==-

Jambase Rules   
-==-==-==-==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==--==-==-  
As obj.o : source.s ;   
 Assemble the file source.s. Called by the Object rule.

Bulk directory : sources ;   
 Copies sources into directory.

Cc object : source ;   
 Compile the file source into object, using the C compiler $(CC),   
 its flags $(CCFLAGS) and $(OPTIM), and the header file directories   
 $(HDRS). Called by the Object rule.  
   
C++ obj.o : source.cc ;   
 Compile the C++ source file source.cc. Called by the Object rule.

Chmod target ;   
 (Unix and VMS only.) Change file permissions on target to target-specific  
 $(MODE) value set by Link, File, Install\*, and Shell rules.

Clean clean : targets ;   
 Removes existing targets when clean is built. clean is not a dependency of  
 all, and must be built explicitly for targets to be removed.

FDefines defines ;   
 Expands a list of definitions into a list of compiler (or preprocessor)   
 switches (such as -Dsymbol=val on Unix) to pass the definitions.

File target : source ;   
 Copies source into target.

FIncludes dirs ;   
 Expands a list of directories into a list of compiler (or preprocessor)   
 switches (such as -Idir on Unix) to add the directories to the header inclusion search path.   
Fortran obj.o : source.f ;   
Compile the Fortran source file source.f. Called by the Object rule.   
FQuote files ;   
Returns each of files suitably quoted so as to hide shell metacharacters (such as whitespace and filename matching wildcards) from the shell.   
GenFile target : image sources ;

Runs the command "image target sources" to create target from sources and image. (where image is an executable built by the Main rule.)   
HardLink target : source ;   
Makes target a hard link to source, if it isn't one already. (Unix only.)   
HdrRule source : headers ;   
Arranges the proper dependencies when the file source includes the files headers through the "#include" C preprocessor directive.   
This rule is not intended to be called explicitly. It is called automatically during header scanning on sources handled by the Object rule (e.g., sources in Main or Library rules).

InstallBin dir : sources ;   
Copy sources into dir with mode $(EXEMODE).   
InstallLib dir : sources ;   
Copy sources into dir with mode $(FILEMODE).   
InstallMan dir : sources ;   
Copy sources into the appropriate subdirectory of dir with mode $(FILEMODE). The subdirectory is mans, where s is the suffix of each of sources.   
InstallShell dir : sources ;   
Copy sources into dir with mode $(SHELLMODE).   
Lex source.c : source.l ;   
Process the lex(1) source file source.l and rename the lex.yy.c to source.c. Called by the Object rule.   
Library library : sources ;   
Compiles sources and archives them into library. The intermediate objects are deleted. Calls Objects and LibraryFromObjects.   
If Library is invoked with no suffix on library, the $(SUFLIB) suffix is used.

LibraryFromObjects library : objects ;   
Archives objects into library. The objects are then deleted.   
If library has no suffix, the $(SUFLIB) suffix is used.

Link image : objects ;   
Links image from objects and sets permissions on image to $(EXEMODE). Image must be actual filename; suffix is not supplied. Called by Main.   
LinkLibraries image : libraries ;   
Makes image depend on libraries and includes them during the linking.   
Image may be referenced without a suffix in this rule invocation; LinkLibraries supplies the suffix.

Main image : sources ;   
Compiles sources and links them into image. Calls Objects and MainFromObjects.   
Image may be referenced without a suffix in this rule invocation; Main supplies the suffix.

MainFromObjects image : objects ;   
Links objects into image. Dependency of exe. MainFromObjects supplies the suffix on image filename.   
MakeLocate target : dir ;   
Creates dir and causes target to be built into dir.   
MkDir dir ;   
Creates dir and its parent directories.   
Object object : source ;   
Compiles a single source file source into object. The Main and Library rules use this rule to compile source files.   
Causes source to be scanned for "#include" directives and calls HdrRule to make all included files dependedencies of object.

Calls one of the following rules to do the actual compiling, depending on the suffix of source:

       \*.c:   Cc   
       \*.cc:  C++   
       \*.cpp: C++  
       \*.C:   C++   
       \*.l:   Lex   
       \*.y:   Yacc  
       \*.\*:   UserObject

ObjectC++Flags source : flags ;   
ObjectCcFlags source : flags ;   
Add flags to the source-specific value of $(CCFLAGS) or $(C++FLAGS) when compiling source. Any file suffix on source is ignored.   
ObjectDefines object : defines ;   
Adds preprocessor symbol definitions to the (gristed) target-specific $(CCDEFS) for the object.   
ObjectHdrs source : dirs ;   
Add dirs to the source-specific value of $(HDRS) when scanning and compiling source. Any file suffix on source is ignored.   
Objects sources ;   
For each source file in sources, calls Object to compile the source file into a similarly named object file.   
RmTemps targets : sources ;   
Marks sources as temporary with the TEMPORARY rule, and deletes sources once targets are built. Must be the last rule invoked on targets. Used internally by LibraryFromObjects rule.   
Setuid images ;   
Sets the setuid bit on each of images after linking. (Unix only.)   
SoftLink target : source ;   
Makes target a symbolic link to source, if it isn't one already. (Unix only.)   
SubDir VAR d1 ... dn ;   
Sets up housekeeping for the source files located in $(VAR)/d1/.../dn:   
Reads in rules file associated with VAR, if it hasn't already been read.   
Initializes variables for search paths, output directories, compiler flags, and grist, using d1 ... dn tokens.   
VAR is the name of a variable; d1 thru dn are elements of a directory path.

SubDirC++Flags flags ;   
SubDirCcFlags flags ;   
Adds flags to the compiler flags for source files in SubDir's directory.   
SubDirHdrs d1 ... dn ;   
Adds the path d1/.../dn/ to the header search paths for source files in SubDir's directory. d1 through dn are elements of a directory path.   
SubInclude VAR d1 ... dn ;   
Reads the Jamfile in $(VAR)/d1/.../dn/.   
Shell image : source ;   
Copies source into the executable sh(1) script image. Ensures that the first line of the script is $(SHELLHEADER) (default #!/bin/sh).   
Undefines images : symbols ;   
Adds flags to mark symbols as undefined on link command for images. Images may be referenced unsuffixed; the Undefines rule supplies the suffix.   
UserObject object : source ;   
This rule is called by Object for source files with unknown suffixes, and should be defined in Jamrules with a user-provided rule to handle the source file types not handled by the Object rule. The Jambase UserObject rule merely issues a complaint when it encounters source with files suffixes it does not recognize.   
Yacc source.c : source.y ;   
Process the yacc(1) file source.y and renamed the resulting y.tab.c and y.tab.h to source.c. Produces a y.tab.h and renames it to source.h. Called by the Object rule.

--------------------------------------------------------------------------------

Jambase Pseudotargets   
There are two kinds of Jam targets: file targets and pseudotargets. File targets are objects that can be found in the filesystem. Pseudotargets are symbolic, and usually represent other targets. Most Jambase rules that define file targets also define pseudotargets which are dependent on types of file targets. The Jambase pseudotargets are:

exe  Executables linked by the Main or MainFromObjects rules    
lib  Libraries created by the Library or LibraryFromObjects rules    
obj  Compiled objects used to create Main or Library targets    
dirs  Directories where target files are written    
file  Files copied by File and Bulk rules    
shell  Files copied by Shell rule    
clean  Removal of built targets (except files copied by Install\* rules)    
install  Files copied by Install\* rules    
uninstall  Removal of targets copied by Install\* rules

In addition, Jambase makes the jam default target "all" depend on "exe", "lib", "obj", "files", and "shell".

--------------------------------------------------------------------------------

Jambase Variables   
Most of the following variables have default values for each platform; refer to the Jambase file to see what those defaults are.

ALL\_LOCATE\_TARGET

Alternative location of built targets. By default, Jambase rules locate built targets in the source tree. By setting $(ALL\_LOCATE\_TARGET) in Jamrules, you can cause jam to write built targets to a location outside the source tree.   
AR   
The archive command used to update Library and LibraryFromObjects targets.   
AS   
The assembler for As rule targets.   
ASFLAGS   
Flags handed to the assembler for As.   
AWK   
The name of awk interpreter, used when copying a shell script for the Shell rule.   
BCCROOT   
Selects Borland compile and link actions on NT.   
BINDIR   
Not longer used. (I.e., used only for backward compatibility with the obsolete INSTALLBIN rule.)   
CC   
C compiler used for Cc rule targets.   
CCFLAGS   
Compile flags for Cc rule targets. The Cc rule sets target-specific $(CCFLAGS) values on its targets.   
C++   
C++ compiler used for C++ rule targets.   
C++FLAGS   
Compile flags for C++ rule targets. The C++ rule sets target-specific $(C++FLAGS) values on its targets.   
CHMOD   
Program (usually chmod(1)) used to set file permissions for Chmod rule.   
CP   
The file copy program, used by File and Install\* rules.   
CRELIB   
If set, causes the Library rule to invoke the CreLib rule on the target library before attempting to archive any members, so that the library can be created if needed.   
CW   
On Macintosh, the root of the Code Warrior Pro 5 directory.   
DEFINES   
Preprocessor symbol definitions for Cc and C++ rule targets. The Cc and C++ rules set target-specific $(CCDEFS) values on their targets, based on $(DEFINES). (The "indirection" here is required to support compilers, like VMS, with baroque command line syntax for setting symbols).   
DOT   
The operating system-specific name for the current directory.   
DOTDOT   
The operating system-specific name for the parent directory.   
EXEMODE   
Permissions for executables linked with Link, Main, and MainFromObjects, on platforms with a Chmod action.   
FILEMODE   
Permissions for files copied by File or Bulk, on platforms with a Chmod action.   
FORTRAN   
The Fortran compiler used by Fortran rule.   
FORTRANFLAGS   
Fortran compiler flags for Fortran rule targets.   
GROUP   
(Unix only.) The group owner for Install\* rule targets.   
HDRGRIST   
If set, used by the HdrRule to distinguish header files with the same name in diffrent directories.   
HDRPATTERN   
A regular expression pattern that matches C preprocessor "#include" directives in source files and returns the name of the included file.   
HDRRULE   
Name of the rule to invoke with the results of header file scanning. Default is "HdrRule".   
This is a jam-special variable. If both HDRRULE and HDRSCAN are set on a target, that target will be scanned for lines matching $(HDRSCAN), and $(HDDRULE) will be invoked on included files found in the matching $(HDRSCAN) lines.

HDRS   
Directories to be searched for header files. This is used by the Object rule to:   
set up search paths for finding files returned by header scans   
add -I flags on compile commands   
(See STDHDRS.)   
HDRSCAN   
Regular expression pattern to use for header file scanning. The Object rule sets this to $(HDRPATTERN). This is a jam-special variable; see HDRRULE.   
HDRSEARCH   
Used by the HdrRule to fix the list of directories where header files can be found for a given source file.   
INSTALLGRIST   
Used by the Install\* rules to grist paths to installed files; defaults to "installed".   
JAMFILE   
Default is "Jamfile"; the name of the user-written rules file found in each source directory.   
JAMRULES   
Default is "Jamrules"; the name of a rule definition file to be read in at the first SubDir rule invocation.   
KEEPOBJS   
If set, tells the LibraryFromObjects rule not to delete object files once they are archived.   
LEX   
The lex(1) command and flags.   
LIBDIR   
Not longer used. (I.e., used only for backward compatibility with the obsolete INSTALLLIB rule.)   
LINK   
The linker. Defaults to $(CC).   
LINKFLAGS   
Flags handed to the linker. Defaults to $(CCFLAGS).   
LINKLIBS   
List of external libraries to link with. The target image does not depend on these libraries.   
LN   
The hard link command for HardLink rule.   
LOCATE\_SOURCE   
Used to set the location of generated source files. The Yacc, Lex, and GenFile rules set LOCATE on their targets to $(LOCATE\_SOURCE). $(LOCATE\_SOURCE) is initialized by the SubDir rule to the source directory itself. (Also, see ALL\_LOCATE\_TARGET.)   
LOCATE\_TARGET   
Used to set the location of built binary targets. The Object rule, and hence the Main and Library rules, set LOCATE on their targets to $(LOCATE\_TARGET). $(LOCATE\_TARGET) is initialized by the SubDir rule to the source directory itself. (See ALL\_LOCATE\_TARGET.)   
MANDIR   
Not longer used. (I.e., used only for backward compatibility with the obsolete INSTALLMAN rule.)   
MKDIR   
The 'create directory' command used for the MkDir rule.   
MODE   
The target-specific file mode (permissions) for targets of the Shell, Setuid, Link, and Install\* rules. Used by the Chmod action; hence relevant to NT and VMS only.   
MSVC   
Selects Microsoft Visual C 16-bit compile & link actions on NT.   
MSVCNT   
Selects Microsoft Visual C NT compile & link actions on NT.   
MV   
The file rename command and options.   
NEEDLIBS   
The list of libraries used when linking an executable. Used by the Link rule.   
NOARSCAN   
If set, indicates that library members' timestamps can't be found, and prevents the individual objects from being deleted, so that their timestamps can be used instead.   
NOARUPDATE   
If set, indicates that libraries can't be updated, but only created whole.   
OPTIM   
The C compiler flag for optimization, used by Cc and C++ rules.   
OSFULL   
The concatenation of $(OS)$(OSVER)$(OSPLAT), used when jam builds itself to determine the target binary directory. $(OS) and $(OSPLAT) are determined by jam at its compile time (in jam.h). $(OSVER) can optionally be set by the user.   
OWNER   
The owner of installed files. Used by Install\* rules.   
RANLIB   
The name of the ranlib command. If set, causes the Ranlib action to be applied after the Archive action to targets of the Library rule.   
RELOCATE   
If set, tells the Cc rule to move the output object file to its target directory because the cc command has a broken -o option.   
RM   
The command and options to remove a file.   
SEARCH\_SOURCE   
The directory to find sources listed with Main, Library, Object, Bulk, File, Shell, InstallBin, InstallLib, and InstallMan rules. This works by setting the jam-special variable SEARCH to the value of $(SEARCH\_SOURCE) for each of the rules' sources. The SubDir rule initializes SEARCH\_SOURCE for each directory.   
SHELLHEADER   
A string inserted to the first line of every file created by the Shell rule.   
SHELLMODE   
Permissions for files installed by Shell rule.   
SOURCE\_GRIST   
Set by the SubDir to a value derived from the directory name, and used by Objects and related rules as 'grist' to perturb file names.   
STDHDRS   
Directories where headers can be found without resorting to using the flag to the C compiler. The $(STDHDRS) directories are used to find headers during scanning, but are not passed to the compiler commands as -I paths.   
SUBDIR   
The path from the current directory to the directory last named by the SubDir rule.   
TOP   
The path from the current directory to the directory that has the Jamrules file. Used by the SubDir rule.   
SUFEXE   
The suffix for executable files, if none provided. Used by the Main rule.   
SUFLIB   
The suffix for libraries. Used by the Library and related rules.   
SUFOBJ   
The suffix for object files. Used by the Objects and related rules.   
UNDEFFLAG   
The flag prefixed to each symbol for the Undefines rule (i.e., the compiler flag for undefined symbols).   
WATCOM   
Selects Watcom compile and link actions on OS2.   
YACC   
The yacc(1) command.   
YACCFILES   
The base filename generated by yacc(1).   
YACCFLAGS   
The yacc(1) command flags.   
YACCGEN   
The suffix used on generated yacc(1) output.