# Holle World走入Makefile

Makefile系列

所有编程语言的入门都从Holle World开始，这几乎成了惯例，本文也将以输出Holle World走入Makefile的世界

## Makefile所需环境

环境的建立有多种方式：

1. 在Windows上安装Cgywin
2. 在Windows上安装Linux虚拟机
3. 准备一台Linux卡片机，例如树莓派

对环境进行测试，在命令行窗口输入如下命令

make -v

输出

GNU Make 4.1

Built for x86\_64-pc-linux-gnu

Copyright (C) 1988-2014 Free Software Foundation, Inc.

License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law.

本文的实验环境为在VMware Workstation中建立的ubuntu虚拟机

## 输出Hello World

### 文件编写

现在我们编写一个最简单的Makefile的例子，在命令终端输出Hello World

选择或新建一个目录，在目录中新建一个文件，命名为Makefile，在Makefile文件中输入如下内容

all:

echo "Hello World"

echo前面必须至少有一个TAB键，并且不能出现空格。这种错误很隐蔽，在大多数编辑器中TAB和空格并没有明显区别，所以建议使用可以区分两者的编辑器，比如sublime

### 文件运行

编写完文件之后接下来是运行Makefile，有两种方法运行，一种是在Makefile所在目录运行make命令，第二种是在Makefile所在目录运行make all命令。

ubuntu@liubo:~/my\_project/Makefile\_project/$ make

echo "Hello World"

Hello World

ubuntu@liubo:~/my\_project/Makefile\_project/$ make all

echo "Hello World"

Hello World

到这里我们迈出了万里长征的第一步，我们在命令终端输出了Hello World。

### 对Makefile的改动

#### 改动一

修改之前

all:

echo "Hello World!"

运行结果

ubuntu@liubo:~/my\_project/Makefile\_project/$ make

echo "Hello World"

Hello World

ubuntu@liubo:~/my\_project/Makefile\_project/$ make another

make: \*\*\* No rule to make target 'another'. Stop.

修改之后如下

all:

echo "Hello World!"

another:

echo "show another"

运行结果

ubuntu@liubo:~/my\_project/Makefile\_project/$ make

echo "Hello World!"

Hello World!

ubuntu@liubo:~/my\_project/Makefile\_project/$ make another

echo "show another"

show another

语法总结

从修改和实验的过程我们可以学到以下几点:

1、

### 语法讲解

all:

echo "Hello World"

Makefile的规则：

target... : prerequisites ...

command

...

...

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

echo前面必须至少有一个TAB键，不能出现空格。

## Makefile的文件名

一般Makefile文件取名为“Makefile”、“makefile”或“GNUmakefile”，如果当前目录同时存在三个名字的文件，在执行make命令时，经过测试会优先解析“GNUmakefile”，如果没有“GNUmakefile”会解析“makefile”，最后以上两者都没有才轮到解析“Makefile”。不过一般情况只存在三个文件名中的一个，大部分情况推荐“Makefile”，一些特殊情况make只识别全小写的“makefile”文件名。

当然，也可以使用别的文件名来书写Makefile，比如：“Make.Linux”，“Make.Solaris”，“Make.AIX”等，如果要指定特定的Makefile，可以使用make的“-f”和“--file”参数，如：make -f Make.Linux或make --file Make.AIX。

## Make和Makefile解决的问题

当项目工程非常小时甚至只有一个源文件，直接就可以用gcc命令编译它。但是当项目工程包含很多个源文件时，用gcc命令逐个去编译，就很容易出现混乱而且工作量大。所以我们需要一个工具来管理这些编译过程，那就是make。

make工具类似于批处理程序，可以自动化编译大量源文件，可以使用一条命令实现完全编译，大大提高了编译效率。当项目工程经过一次编译后，在对程序进行修改，根据我们编写的Makefile中的规则和依赖关系，用make进行编译时，可以自动确定哪些文件需要重新编译，只编译修改的文件和对其有依赖关系的文件，而不用再次把所有文件都编译一遍，这样大大节省了时间。

## 涉及的面试问题

make和Makefile的关系是什么？

make是编译项目工程的工具，Makefile是被make解析的文件。输入make命令后查找当前目录的MAkefile文件，然后进行解析并依据解析结果进行编译工作。

## 参考资料

版权声明：本文参考了其他CSDN博主的文章，遵循CC 4.0 BY-SA版权协议，现附上原文出处链接及本声明。

1. Makefile入门一、helloworld 原文链接：https://blog.csdn.net/hjxu2016/article/details/100672126
2. https://www.cnblogs.com/qytan36/archive/2010/05/25/1743955.html