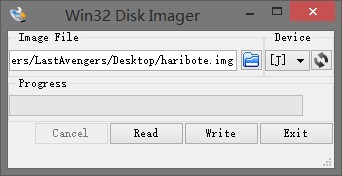
《30天自制操作系统》读书笔记（2）hello, world

* **让系统跑起来**

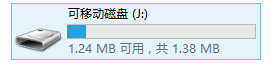
要写一个操作系统，我们首先要有一个储存系统的介质，原版书似乎是06年出版的，可惜那时候没有电脑，没想到作者用的还是软盘，现在的电脑谁有软驱？不得已我使用一张128M的SD卡来代替，而事实上你用的是U盘还是软盘对我们的操作系统没有影响，缺点是你的U盘刷入系统后容量只能是1440 MB，即当年流行的3.5英寸软盘的大小，当然不用担心，再格式化一次（用DiskGeniu），就可以恢复。

我做事情的话，总是怕自己的努力的结果白费了，害怕辛辛苦苦看完这本书但是发现做出来的东西现在根本没法用，比如你花了大力气造了一辆火车，发现轮子的间距和现行标准不符，没轨道可以跑，被标准抛弃的感觉太恐怖，所以我决定试试，作者的系统能不能真正地跑起来。

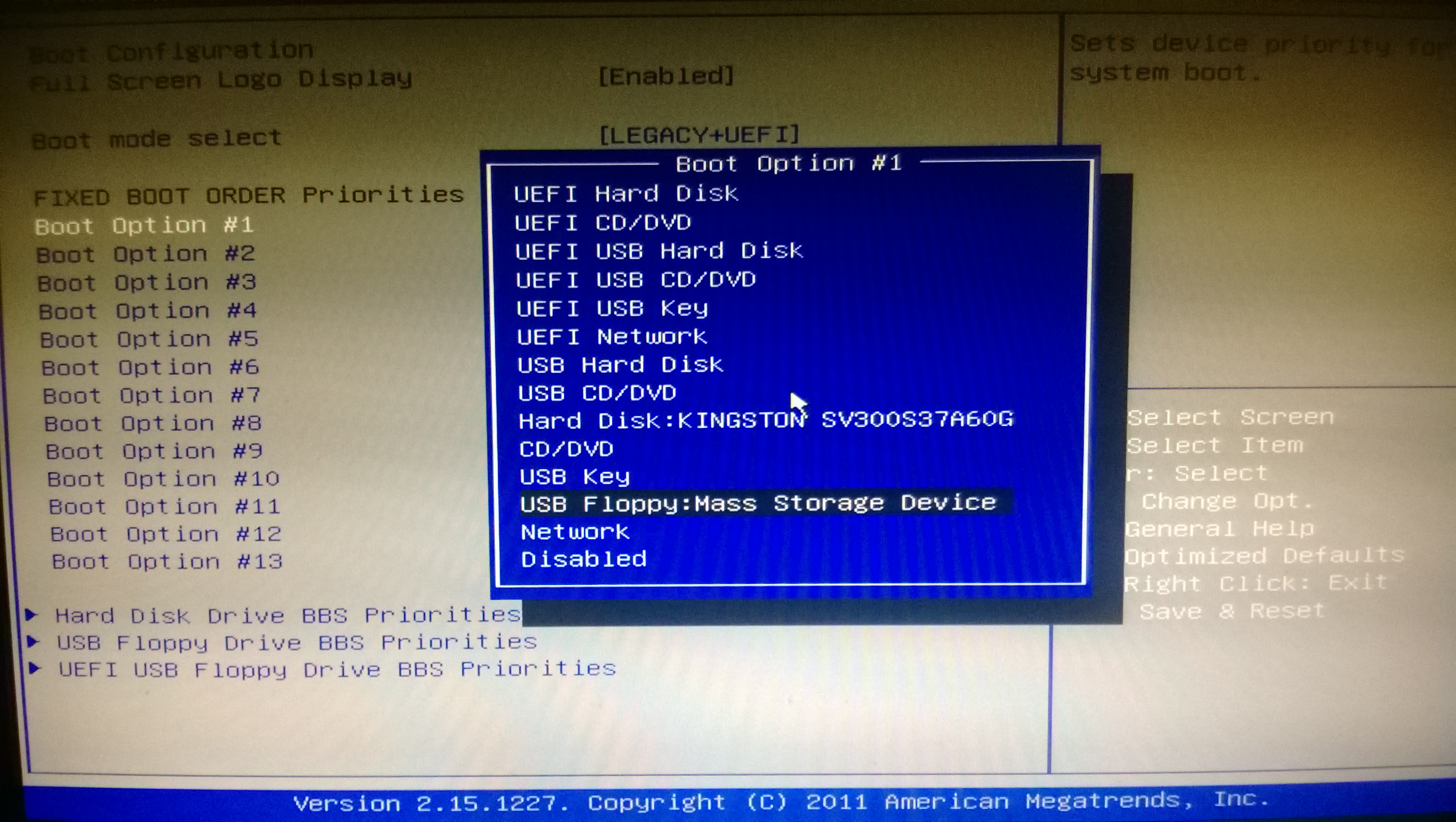
我选择了源码中projects\30\_day\harib27f中的haribote.img文件，用ImgWriter写入到我的储存卡，



写入后的大小和预期相符，1.x MB；



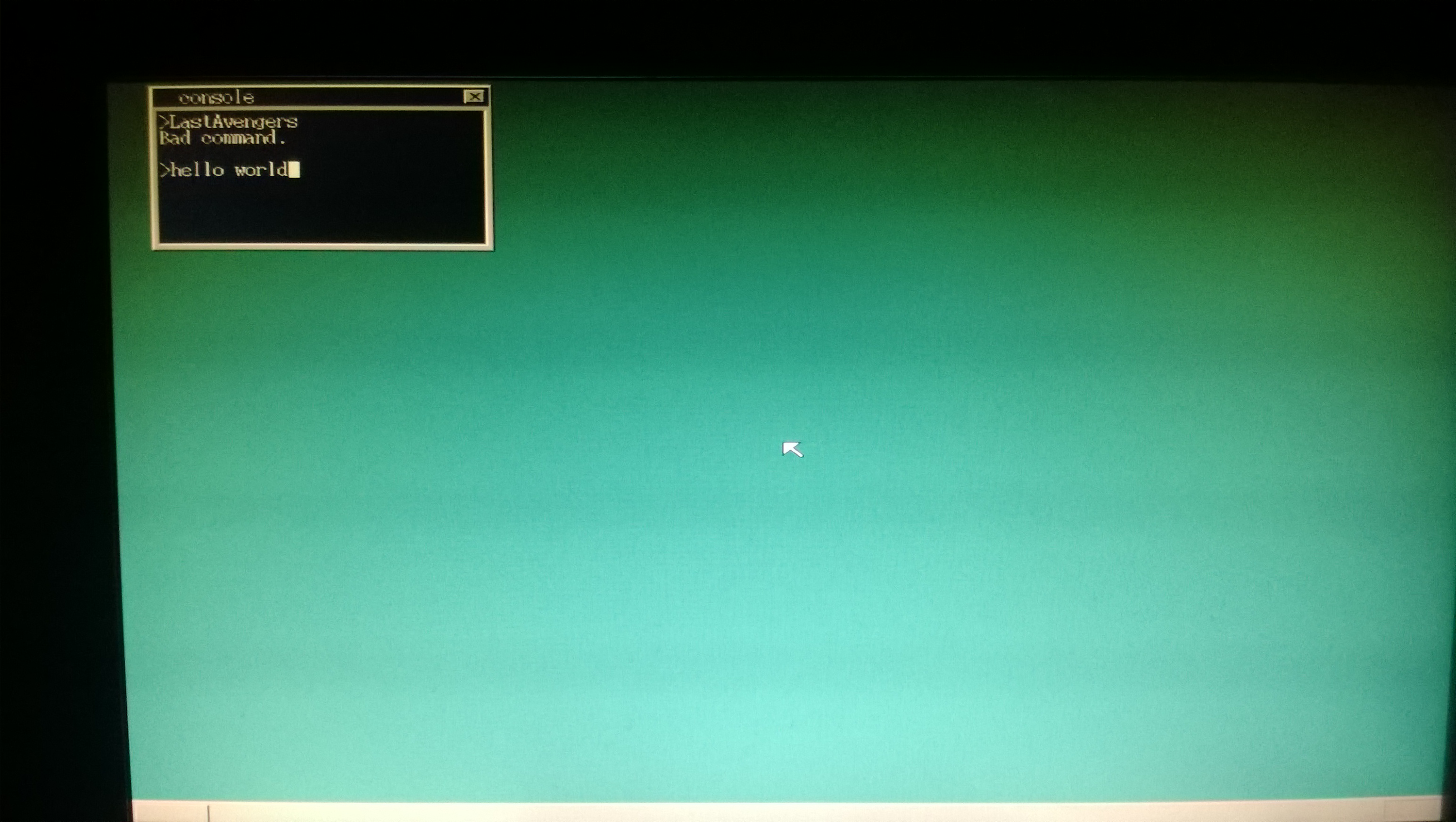
重新启动，开机时狂按Delete，修改启动项；



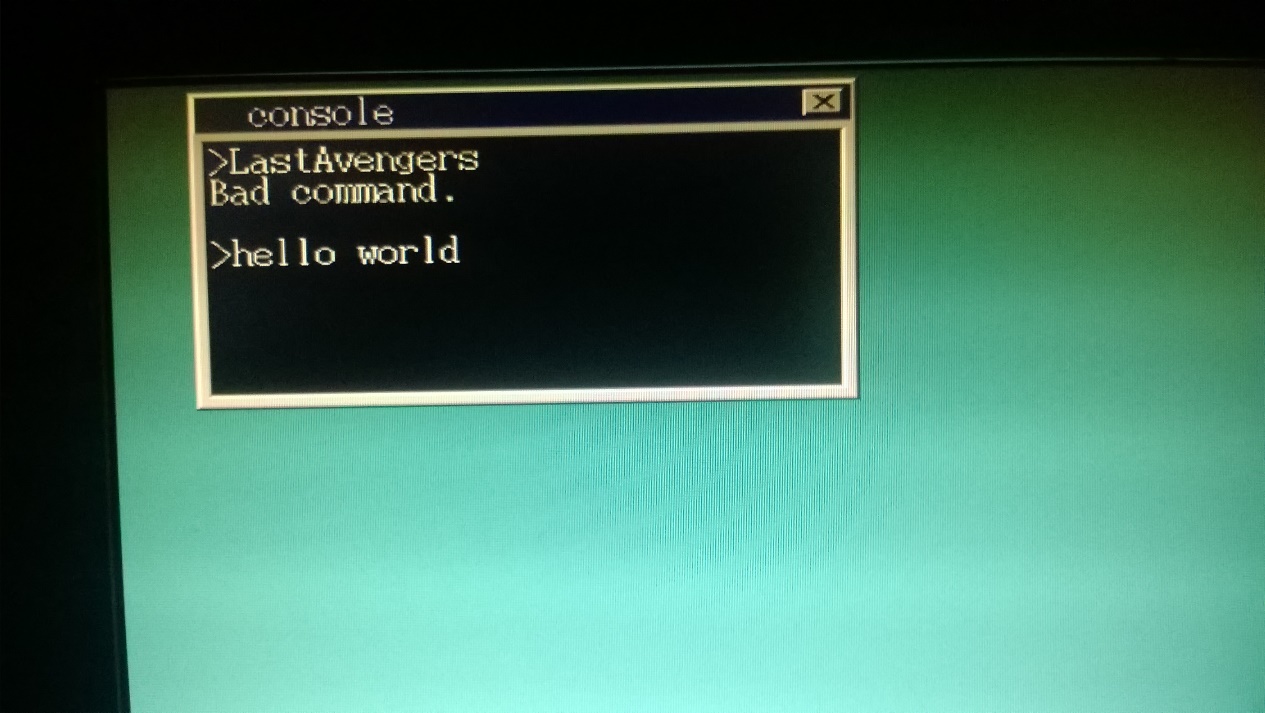
F10保存后重启，心中有些忐忑不安；

… …

但是结果还是如我所愿，看，It works！



这是就是我们要完成的东西了，偷窥胜利果实的快感不言而喻，系统甚至支持USB键盘，但不支持USB鼠标确实是不能用。我试着输入了几个命令：



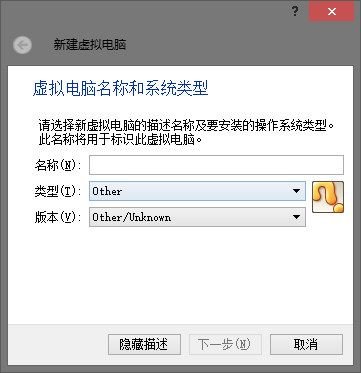
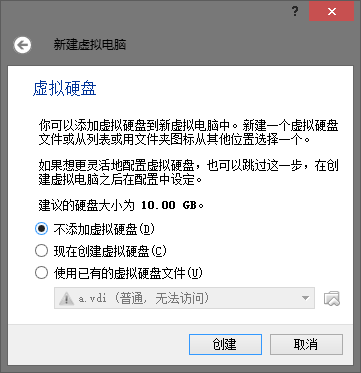
Bad command. 我也不知道什么是作者指定的command，已知的是，exit能用。

* 用Virtual Box 代替物理机

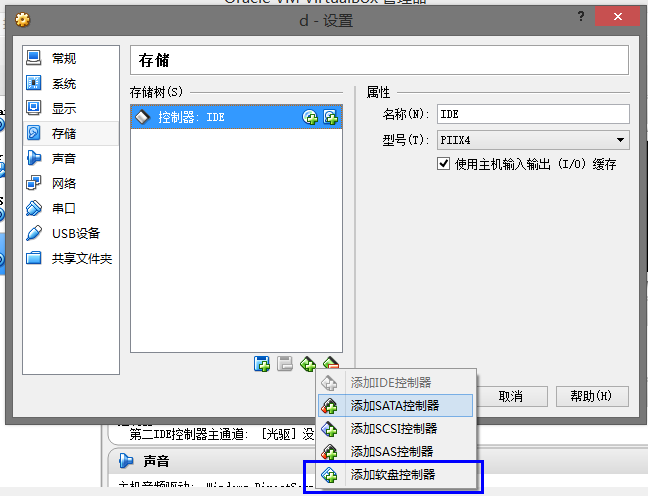
频繁地开关电脑来调试我们的系统是不理想的，我们需要Virtual Box来搭把手。

打开VB的控制台，新建虚拟机，操作系统的类型选Other，版本选择Other\Unknown，

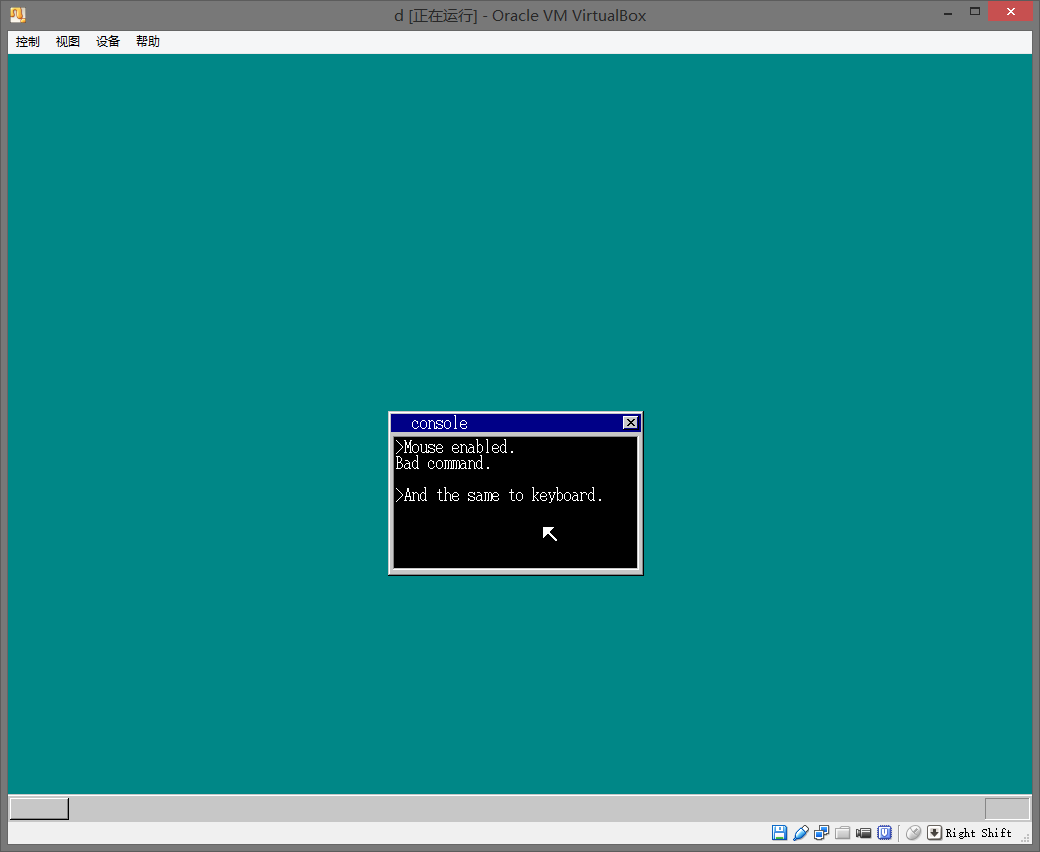
一直点下一步直到虚拟硬盘，选择不添加虚拟硬盘，我们的虚拟硬盘文件就是img。

选择虚拟电脑的设置——储存——储存树，添加一个软盘控制器，原来的IDE控制器可以删掉了，在软盘控制器里新增软盘到控制器，当然选择刚才的haribote.img，然后大功告成，启动系统。



启动系统后一切都显得那么完美，鼠标能用，键盘也是，而且非常方便，输入的蹩脚英文或许有错，见笑了。



PS: 这里需要注意的是本来有另一种方案，使用VB安装目录下的VBoxManage.exe 执行 VBoxManage convertdd  file.img file.vdi

但不知为何，没办法转换上述的haribote.img，只能转换下面要写的hello, world。

* 动手写操作系统

电脑启动的步骤是简要部分步骤是：加电——读取BIOS——自检——控制权移交操作系统（或者说引导），如此看来，我们的任务就是编写一段符合规范的代码，在第四步的时候代码会被执行。

首先我们需要一个标准的FAT12的启动扇区（Boot sector）的代码，我很希望有FAT32 的，无奈这本书给的就是FAT12的，代码如下：

; OS 0.01

; 标准FAT12软盘专用代码

DB 0xeb, 0x4e,0x90 ；不知道什么意思

DB "HELLOIPL" ; 启动扇区(boot sector)的名字

DW 512 ; 每个扇区(sector)的大小:512B

DB 1 ; 簇(cluster)的大小:一个扇区

DW 1 ; FAT的起始位置

DB 2 ; FAT的个数

DW 224 ; 根目录大小

DW 2880 ; 磁盘大小 2880扇区

DB 0xf0 ; 磁盘种类

DW 9 ; FAT长度

DW 18 ; 一个磁道(track)有18个扇区

DW 2 ; 磁头数

DD 0 ; 不使用分区

DD 2880 ; 重写磁盘大小

DB 0,0,,0x29 ; 固定

DD 0xffffffff ; 意义不明

DB "HELLP\_OS " ; 磁盘名称11字节

DB "FAT12 " ; 磁盘的格式名称8字节

RESB 18 ; 空出18 字节

但是这段完全由数据组成的代码只是符合了一个软盘启动扇区的标准，还没有任何可执行的代码，我复制了光盘中的代码，仅仅是修改了显示的文字。

其中有很多我不解的东西。

; hello-os

; TAB=4

ORG 0x7c00

; 将程序加载到地址0x7c00处，已知的是0x7c00-0x7dff是规定的启动区内容的装载地址

JMP entry ; 跳到要执行的代码处

DB 0x90 ；为什么之前的 0xeb 0x4e可以被这上面的三句替代？

DB "HELLOIPL"

. . . ; 这中间的都和上面的代码相同

RESB 18

entry:

MOV AX,0 ; 初始化AX

MOV SS,AX ; 初始化SS，DS，ES，实验发现MOV REG,0 会出错，不知如何解

MOV SP,0x7c00

MOV DS,AX

MOV ES,AX

MOV SI,msg ; 将字符串的首地址复制到源变址寄存器

putloop:

MOV AL,[SI] ; 将字符存到AL

ADD SI,1 ; 移动SI指针指向下一个字符

CMP AL,0 ; 直到读取到0 “DB 0”的时候跳入fin死循环

JE fin

MOV AH,0x0e ; 10H号中断的 0E号功能，已打印机的方式打印一个字符

MOV BX,0x15 ；BH是视频页页码，BL是颜色（仅在图形模式下有效），现在是文字模式？改动BL的值不能改变文字的颜色

INT 0x10 ; 调用10H号中断

JMP putloop

fin:

HLT ；待机

JMP fin

msg:

DB 0x0a, 0x0a ; 回车0AH = #13

DB "LastAvengers's OS"

DB 0x0a

DB 0 ; 结束标志

RESB 0x7dfe-$ ; 这里不应该是0x01fe + 0x7c00 = 0x7dfe

DB 0x55, 0xaa ；启动扇区的最后两个字节必须填充 55AA

DB 0xf0, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00 ; 这后面的删掉也没事，以上是真正的启动区；

RESB 4600

DB 0xf0, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00

RESB 1469432

使用作者提供的nask.exe，执行 nask.exe IPL.asm a.img 可以得到镜像文件，可以利用VB来启动了。

* 简化操作流程：

一开始作者为我们提供了install.bat，!cons\_nt.bat，run.bat来安装和运行系统，后

来又介绍了make.exe，实在是神器，（之前还不解为什么在Linux下编译包需要make install，现在终于知道了），只需要构造一个不带扩展名的MakeFile文件，就可以集编译写入运行于一身，MakeFile 的基本格式如下：

 宏定义  源文件之间的相互依赖关系  
　　 任意可执行的Shell命令

作者在这里贸贸然地给出了

helloos.img : ipl.bin Makefile

../z\_tools/edimg.exe imgin:../z\_tools/fdimg0at.tek \

wbinimg src:ipl.bin len:512 from:0 to:0 imgout:helloos.img

先通过nask 生成bin文件再用edimg.exe 转为img文件，我不知道为什么要这么做，

而且生成 .lst 文件的时候也失败了。

作者给出的makefile中地址都是斜杠，而系统用的是反斜杠，虽然效果一样，但是看着不爽，vim “:1,$/\//\\/g”可以将所有斜杠转化为反斜杠，注意这里的\有的是用来做转义的。

于是我没有照着他那样，我的MakeFile改写如下，同样正常工作：

# nask.exe ipl.asm a.img ipl.lst ::fail NASK: LSTBUF is no enough

default :

tolset\z\_tools\make.exe install

tolset\z\_tools\make.exe run

del \*.\*~ >nul ：：删除临时文件

del \*~ >nul

a.img : ipl.asm Makefile

tolset\z\_tools\nask.exe ipl.asm a.img：：直接编译成img，不知作者的用意是什么

install :

tolset\z\_tools\make.exe -r a.img

run :

echo Running...

"D:\Program Files\Oracle\VirtualBox\VirtualBox.exe" --comment "OS1" --startvm "a5c4b0e6-e142-4720-98ee-056911204b29" ：：虚拟机的快捷方式

echo Finished.

另外改写了！cons\_nt.bat，增加了环境变量。

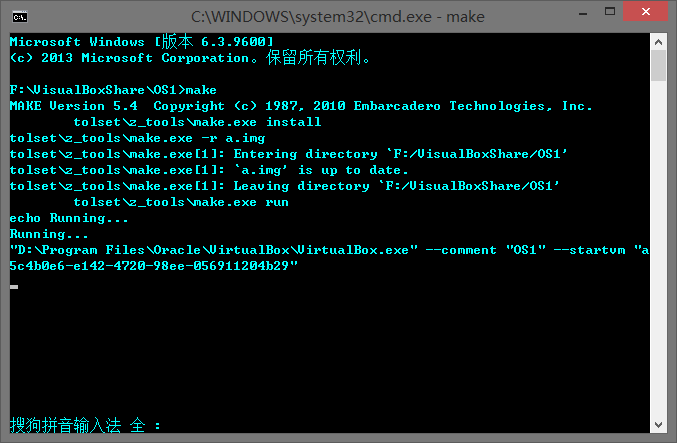
@echo off

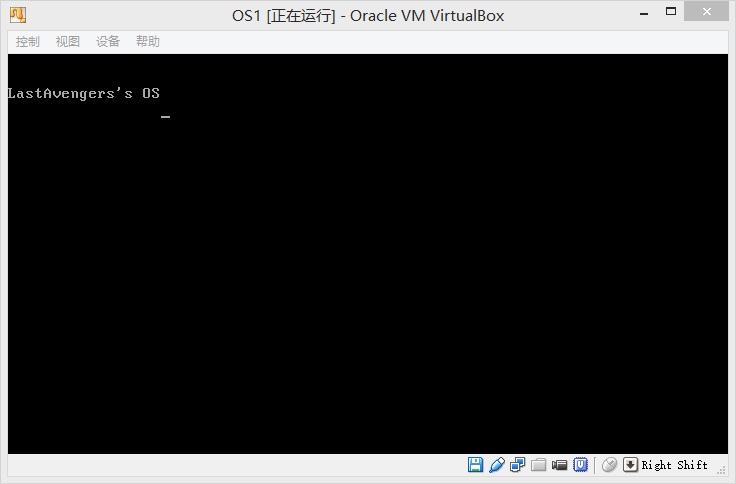
color 0b

set PATH=%PATH%;tolset\z\_tools\

cmd.exe

之后打开!cons\_nt.bat, 输入make，效果如图：





* 知识点：

FAT12启动区的标准：第511个字节开始填充55AA，软盘大小是2880\*512/1024 = 1440 KB;

启动区的加载地址是 0x7c00—0x7dff；

各种寄存器，只有SI，DI，BX为数不多的几个寄存器才能放地址；

MakeFile 的用法；

Vim的替换命令；

int 0x10中断：<http://blog.csdn.net/thimin/article/details/2313390>。

唉，我真是话唠。