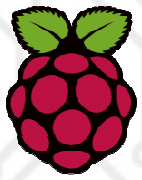
Programming the Raspberry Pi

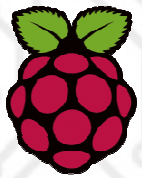
Dr Eben Upton
Raspberry Pi Foundation



element14
www.element14.com

contents

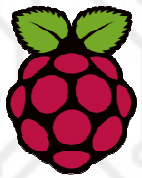
- introduction
- unboxing and setup
- flashing an SD card
- logging in for the first time
- the JOE text editor
- running the “hello world” program
- a (slightly) more complex example
- an OpenGL ES graphics program in C
- the configuration file
- wrap up



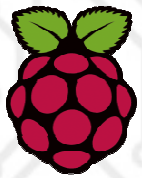
introduction

- Raspberry Pi is a small, cheap ARM-based PC for education and hobbyists
- Runs Debian GNU/Linux from an SD card
  - Standard image available from **http://www.element14.com**
  - Includes a broad range of tools and examples
- General-purpose IO connector allows simple interfacing

Feature	Specification
CPU	700MHz ARM1176-JZFS
GPU	Broadcom VideoCore IV
Memory	256MB LPDDR2-800
Video	HDMI, composite
Audio	HDMI, stereo analog
USB	2 x USB2.0 (model B)
Storage	SD card
Networking	10/100 Ethernet
Power	5V micro USB

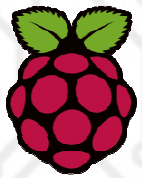


unboxing



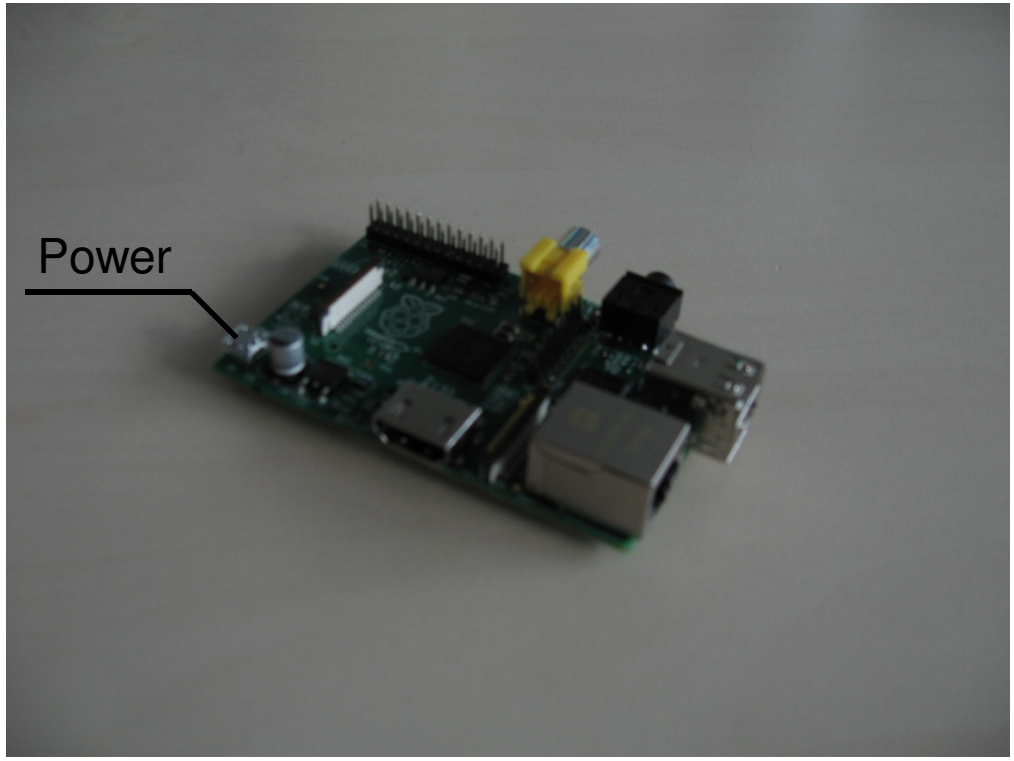
element14
www.element14.com

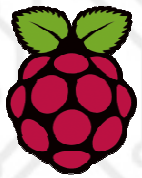
a quick tour



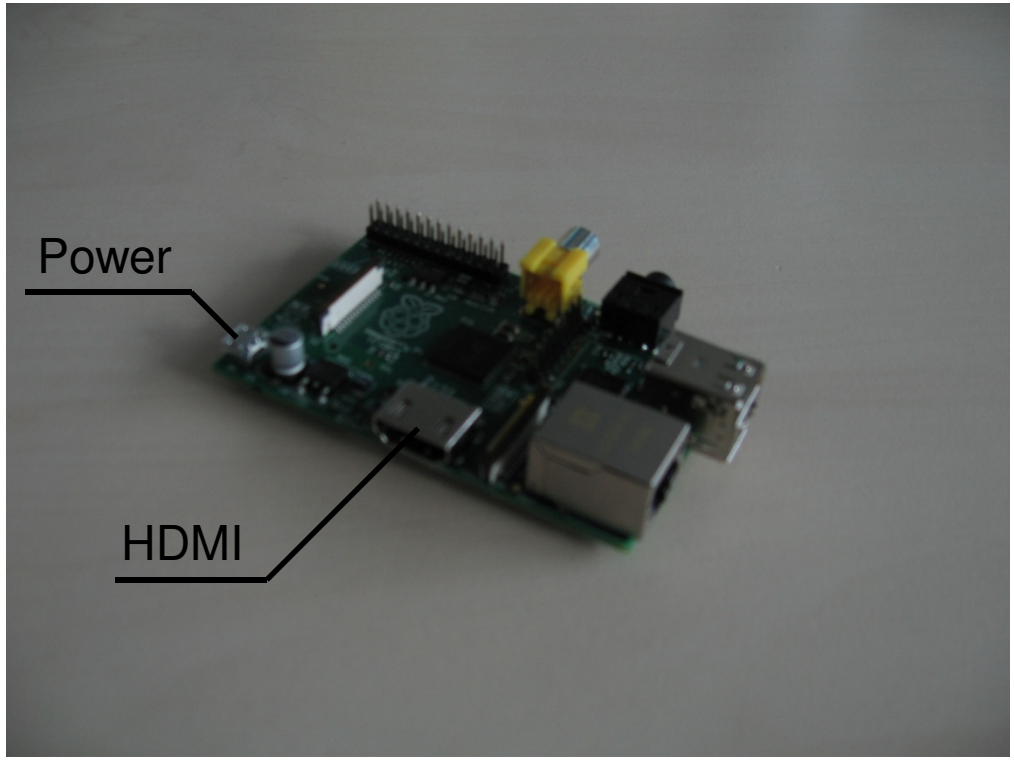
element14
www.element14.com

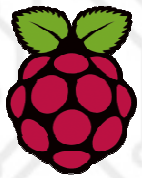
a quick tour



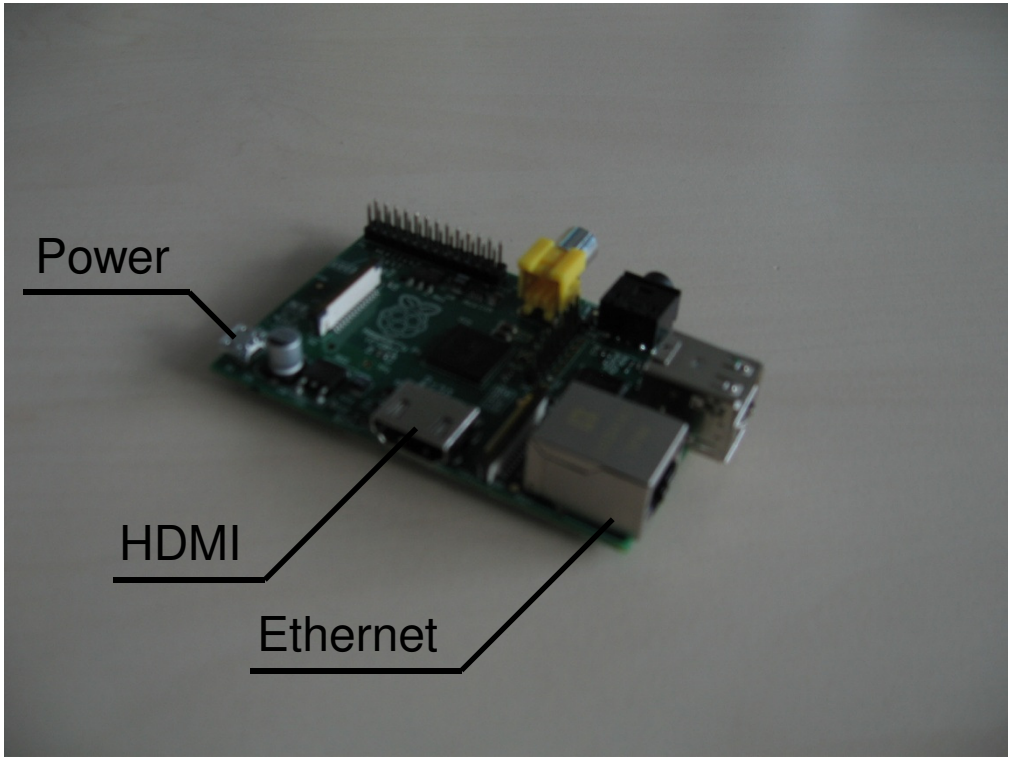


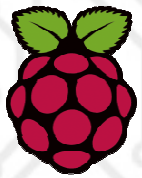
a quick tour



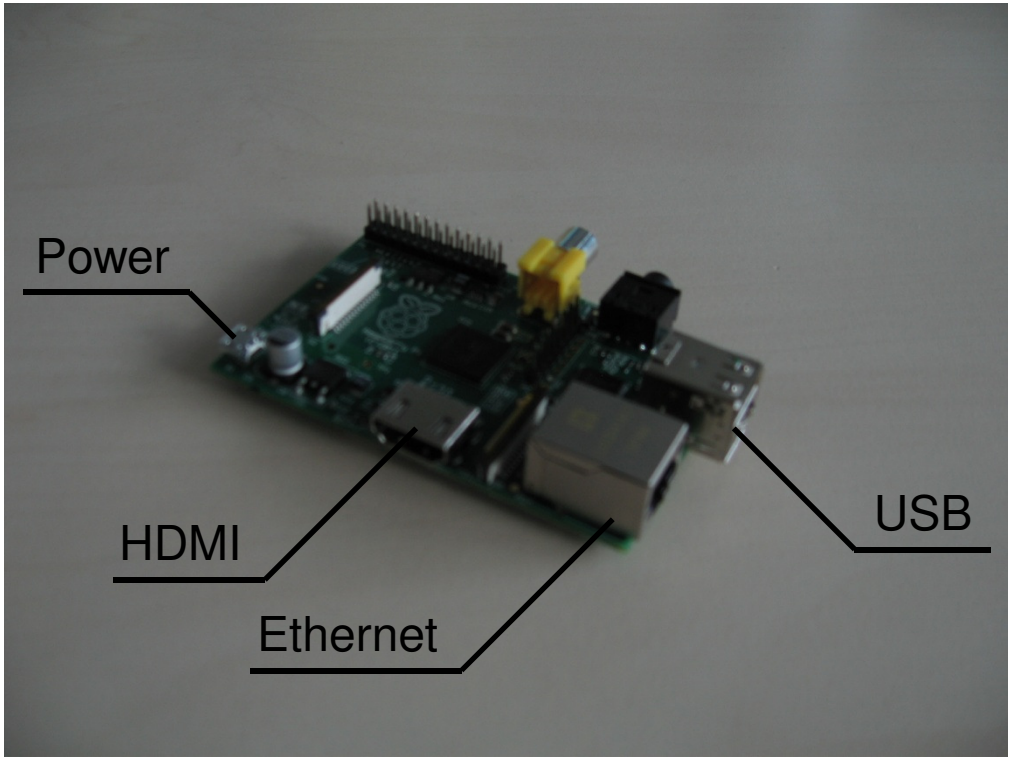


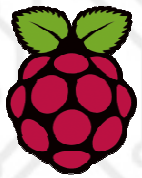
a quick tour



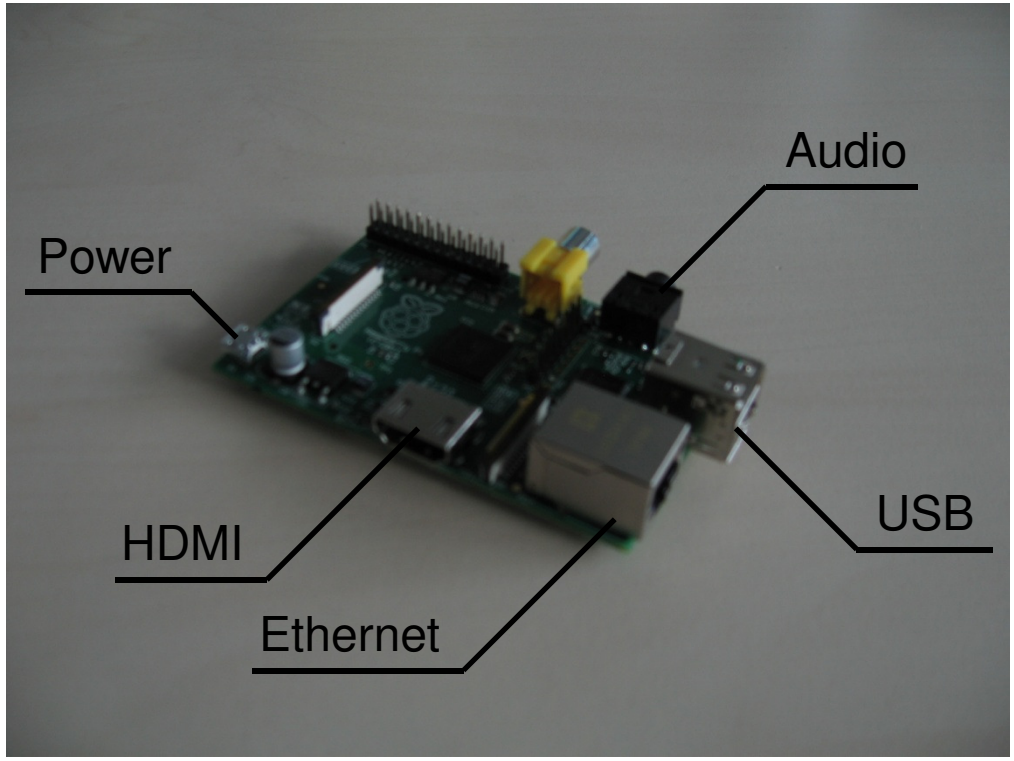


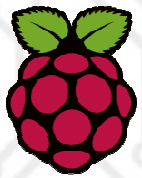
a quick tour



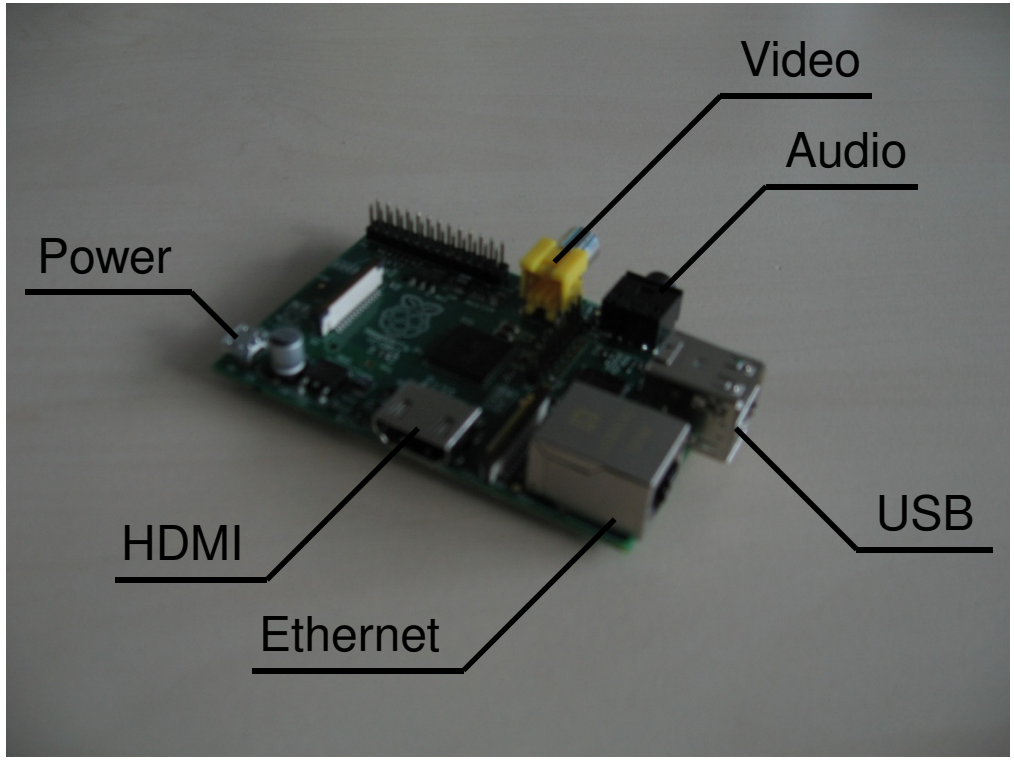


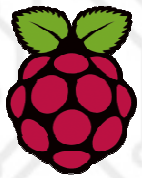
a quick tour



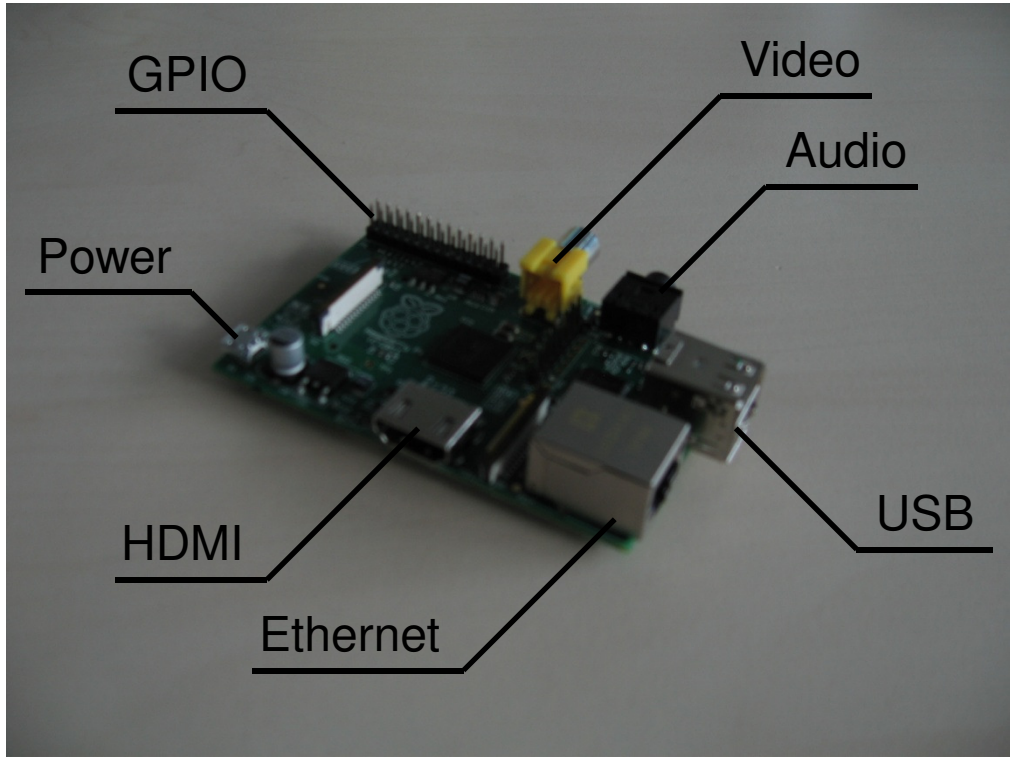


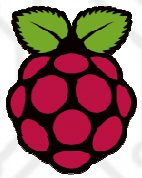
a quick tour



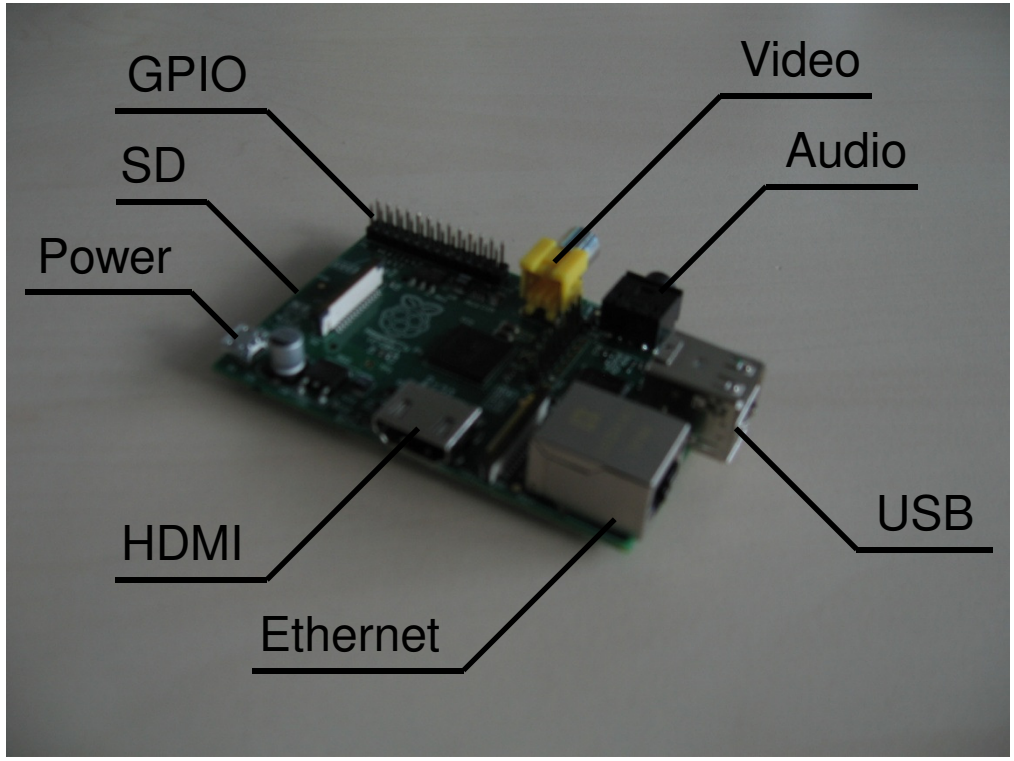


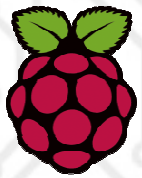
a quick tour



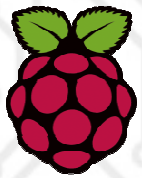


a quick tour



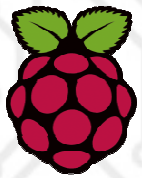


cables and accessories



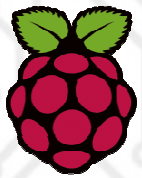
element14
www.element14.com

putting it all together



element14
www.element14.com

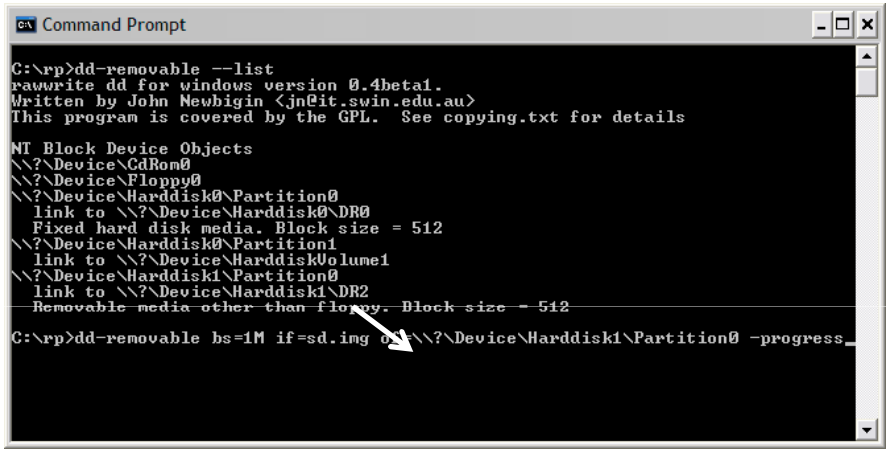
putting it all together

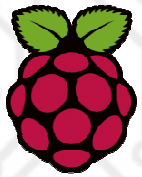


element14
www.element14.com

flashing an SD card

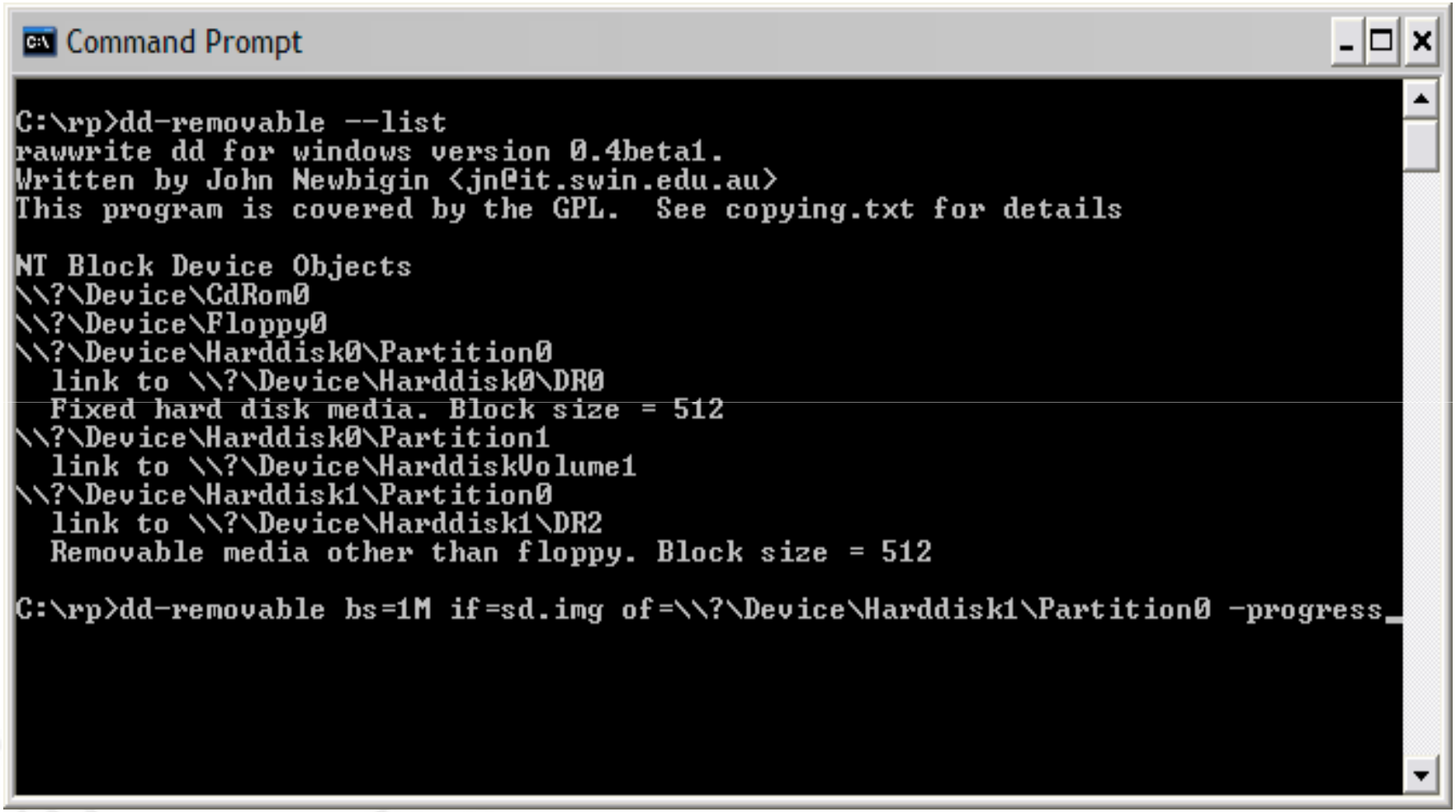
- You may have purchased a pre-installed card
- Otherwise, you will need to
  - Download an image and a copy of the tool **dd-removable** from **www.element14.com/raspberrypi**
  - Flash the image onto a 2GB SD card from a Windows PC
- Insert the card into a card reader
- At a command prompt, type
  - **dd-removable --list**
  - **dd-removable bs=1M if=sd.img of= \\?\Device\Harddisk<X>\Partition0 -progress**
  - Substituting the appropriate number for **<X>**

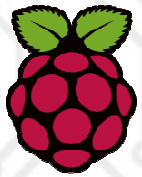




element14
www.element14.com

flashing an SD card

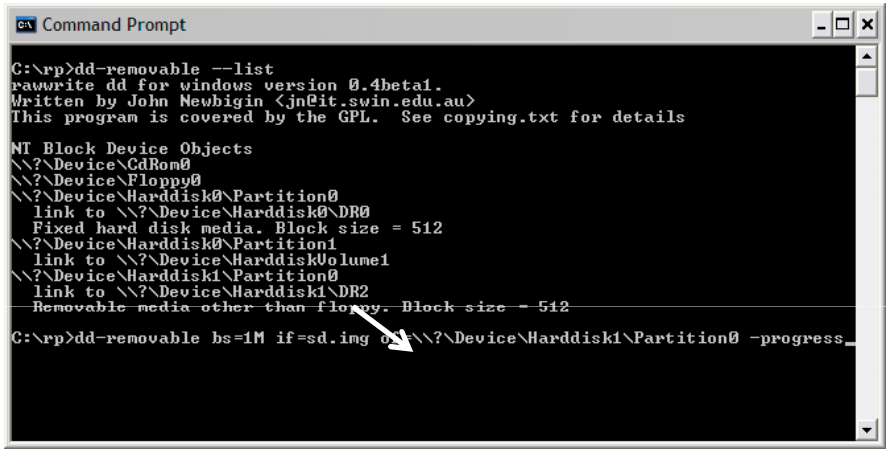


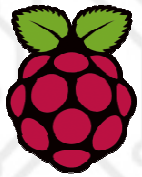


element14
www.element14.com

flashing an SD card

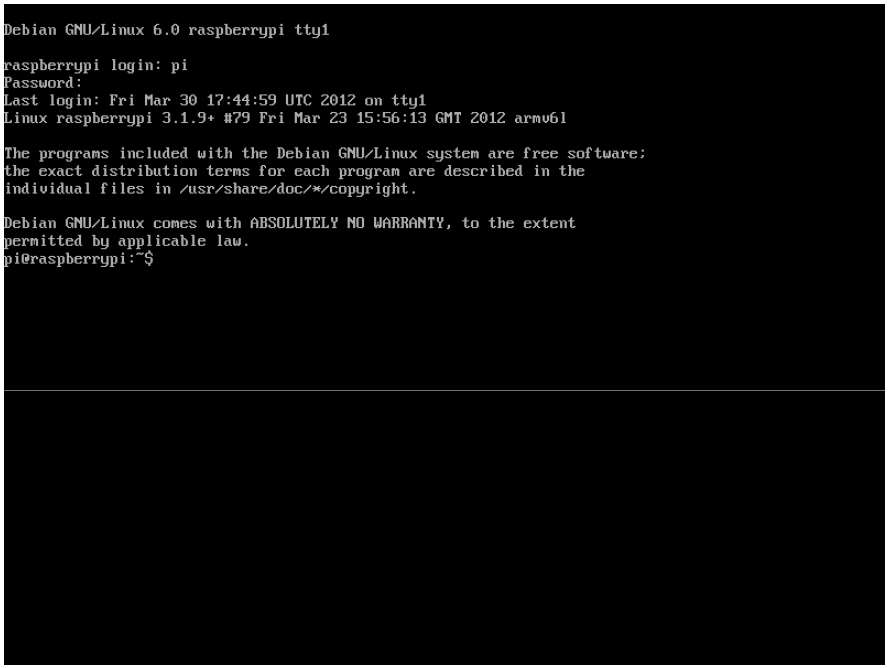
- You may have purchased a pre-installed card
- Otherwise, you will need to
  - Download an image and a copy of the tool **dd-removable** from **www.element14.com/raspberrypi**
  - Flash the image onto a 2GB SD card from a Windows PC
- Insert the card into a card reader
- At a command prompt, type
  - **dd-removable --list**
  - **dd-removable bs=1M if=sd.img of= \\?\Device\Harddisk<X>\Partition0 -progress**
  - Substituting the appropriate number for **<X>**





element14
www.element14.com

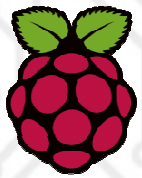
logging in for the first time



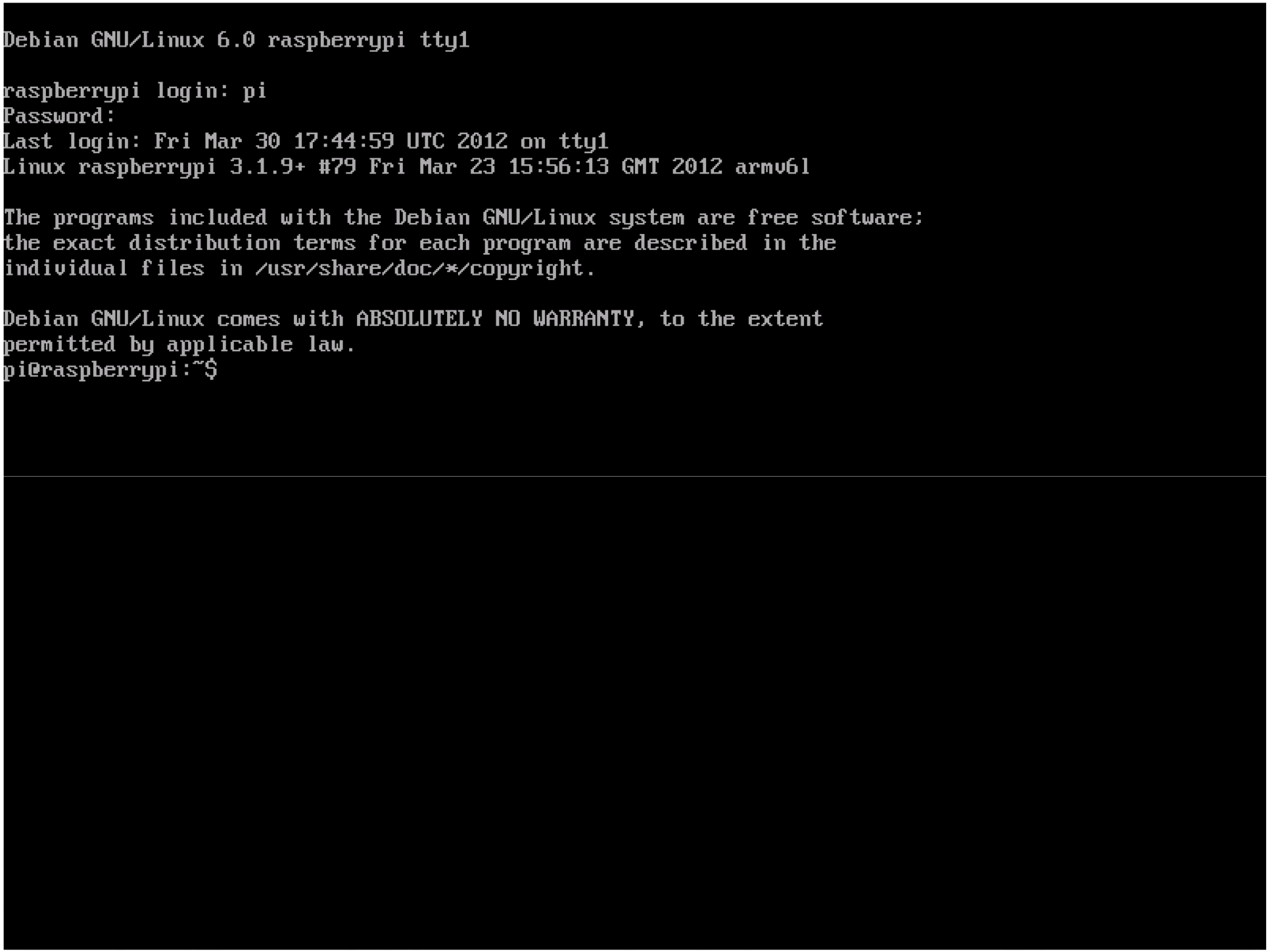
- Insert a card
- Apply power to the device
- Red LED should come on
- After 5 seconds
  - Green LED should begin to flicker
  - Text should appear on the screen
- At the login prompt

```
raspberrypi login:
```

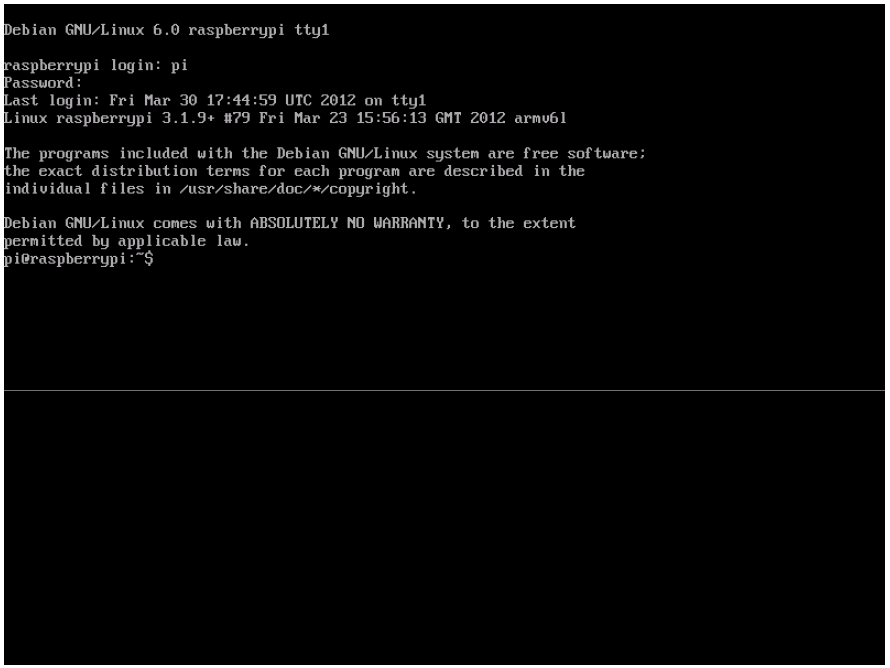
  enter the username **pi**, and password **raspberry**
- You may want to set the clock!



element14
www.element14.com


Debian GNU/Linux 6.0 raspberrypi tty1
raspberrypi login: pi
Password:
Last login: Fri Mar 30 17:44:59 UTC 2012 on tty1
Linux raspberrypi 3.1.9+ #79 Fri Mar 23 15:56:13 GMT 2012 armv6l
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
pi@raspberrypi:~$

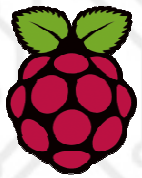
logging in for the first time



- Insert a card
- Apply power to the device
- Red LED should come on
- After 5 seconds
  - Green LED should begin to flicker
  - Text should appear on the screen
- At the login prompt

```
raspberrypi login:
```

  enter the username **pi**, and password **raspberry**
- You may want to set the clock!



element14
www.element14.com

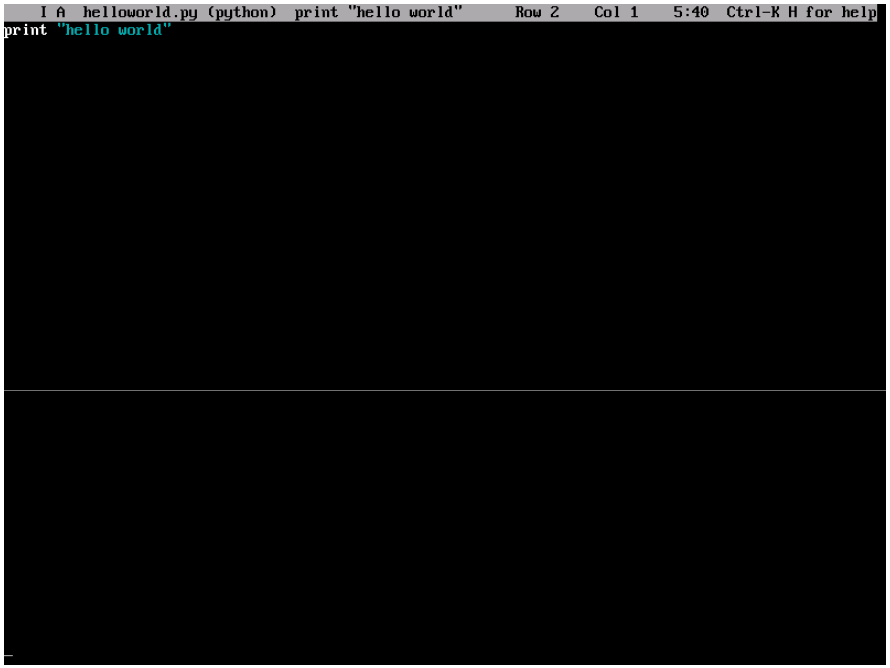
the JOE text editor

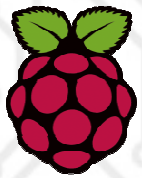
- Standard image bundles JOE
  - Simple programmer's text editor
  - Syntax highlighting for Python and C
- At the command line, type

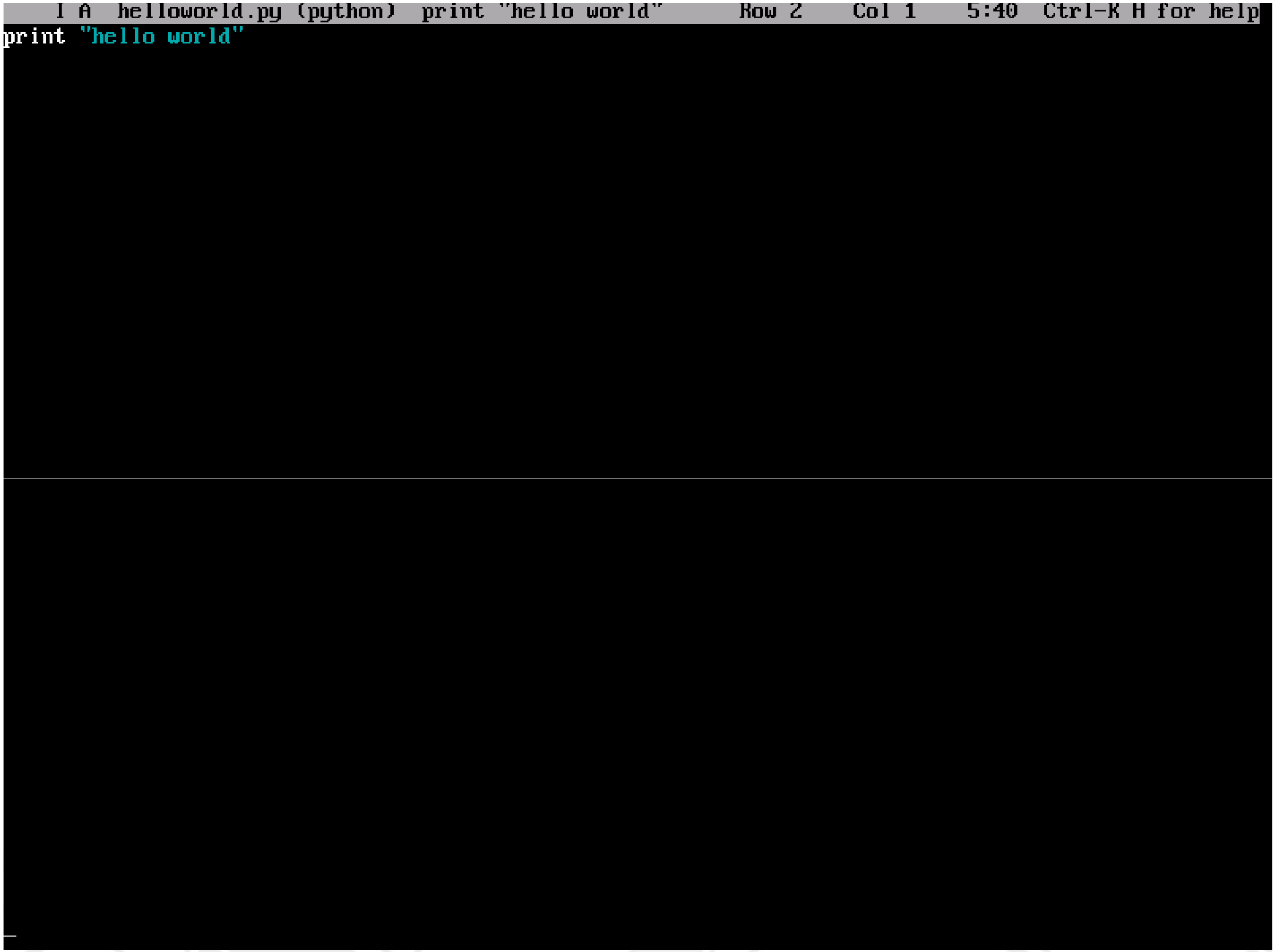
  joe helloworld.py
- When the editor appears, type

  print "hello world"
- Now type Ctrl+K and then X to save and exit
- More documentation available at

  http://joe-editor.sourceforce.net






I A helloworld.py (python) print "hello world" Row 2 Col 1 5:40 Ctrl-K H for help
print "hello world"

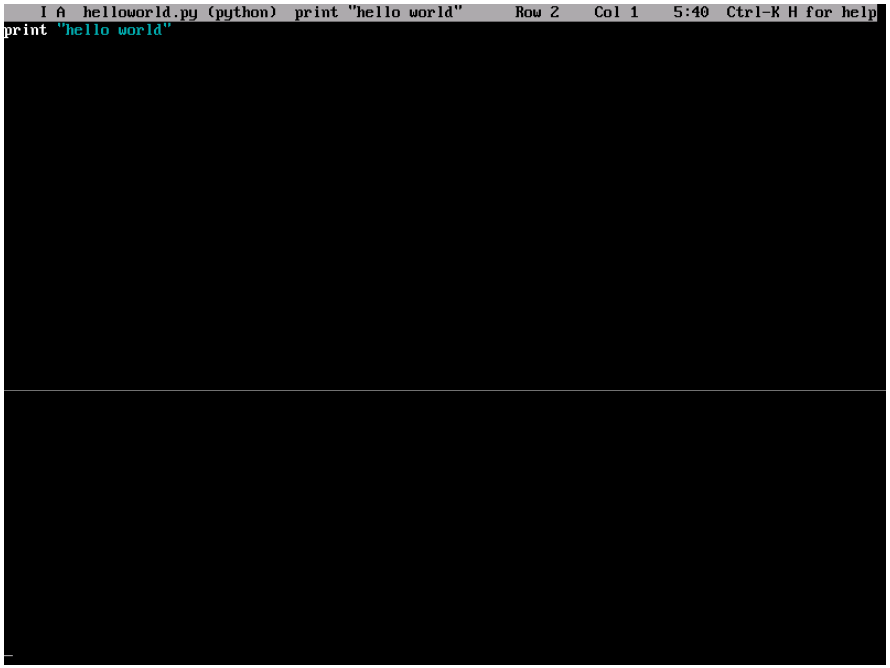
the JOE text editor

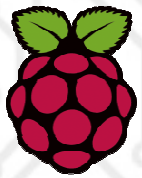
- Standard image bundles JOE
  - Simple programmer's text editor
  - Syntax highlighting for Python and C
- At the command line, type

  joe helloworld.py
- When the editor appears, type

  print "hello world"
- Now type Ctrl+K and then X to save and exit
- More documentation available at

  http://joe-editor.sourceforce.net





running the “hello world” program

- We just wrote our first program!
- We can run it using the bundled Python interpreter
- At the command line, type

  python helloworld.py
- The text “hello world” will appear
- You can also run Python in “interactive mode” by just typing

  python
- A great way to experiment with the language


File helloworld.py saved
pi@raspberrypi:~$ python helloworld.py
hello world
pi@raspberrypi:~$

running the “hello world” program

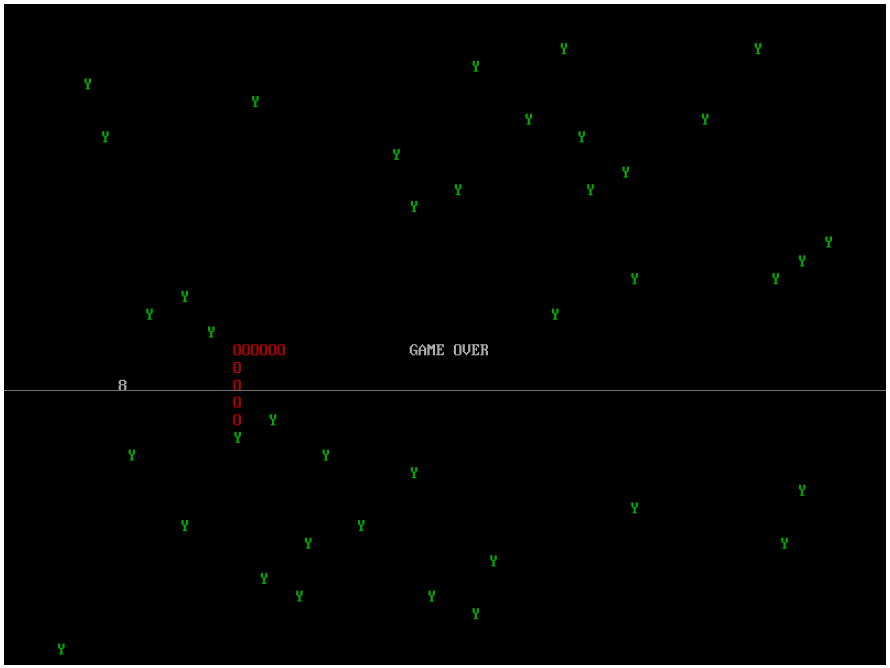
- We just wrote our first program!
- We can run it using the bundled Python interpreter
- At the command line, type

  python helloworld.py
- The text “hello world” will appear
- You can also run Python in “interactive mode” by just typing

  python
- A great way to experiment with the language

a (slightly) more complex program

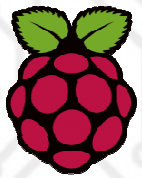
```
  I A  snake.py (Modified)(python)  import random       Row 5    Col 1    5:55  Ctrl-K H for help
import curses
import time
import sys
import random

# function to add a number to an empty place on the screen

def add_number(scr, width, height):
  # loop forever

  while True:
    # make up a random position

    x = random.randint(0, width-1)
    y = random.randint(0, height-1)

    # check if the character at the position is a space

    if scr.inch(y, x) == ord(" "):
      # if it is, replace it with a number and return

      scr.addch(y, x, ord("0") + random.randint(1, 9))

      return

# function to add an obstacle to an empty place on the screen

def add_block(scr, width, height):
  # loop forever

  while True:
    # make up a random position

    x = random.randint(1, width-2)
    y = random.randint(1, height-2)
```



- A series of examples, building up to a simple game of Snake, can be downloaded and unpacked by typing

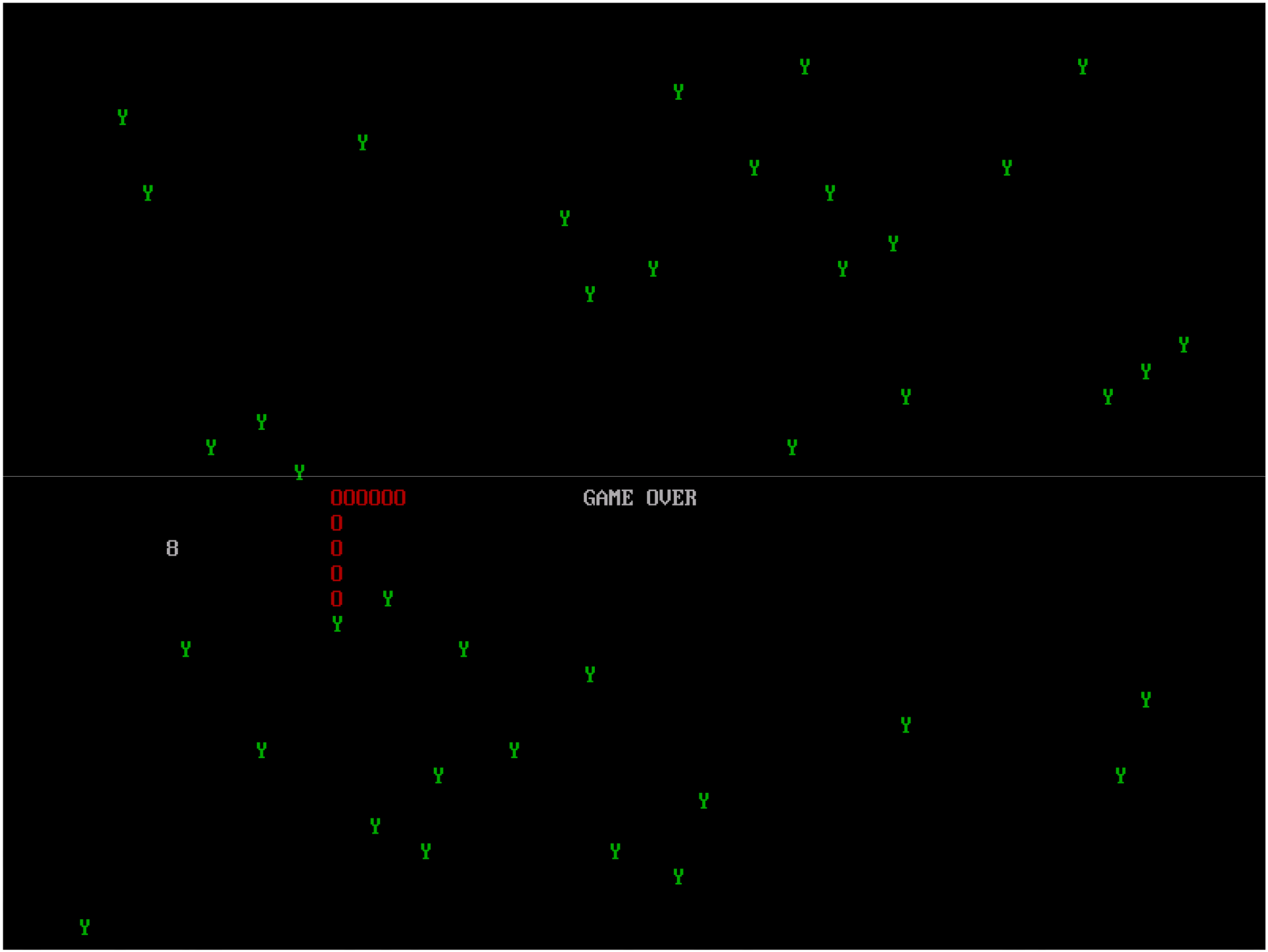
wget http://www.raspberrypi.org/game.tar.gz

tar xvfz game.tar.gz



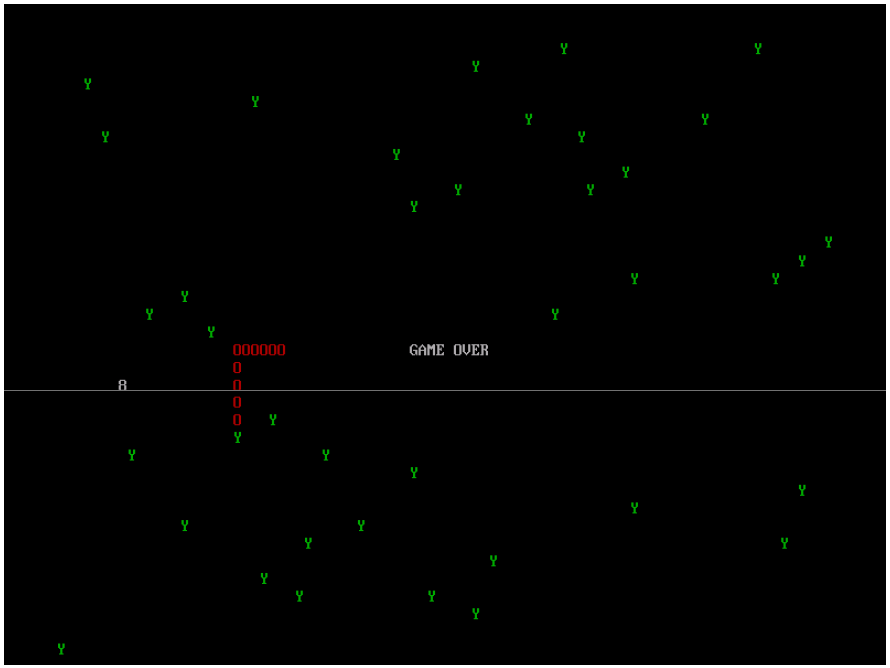
element14
www.element14.com

```
    I A  snake.py (Modified)(python)  import random       Row 5    Col 1    5:55  Ctrl-K H for help
import curses
import time
import sys
import random

# function to add a number to an empty place on the screen

def add_number(scr, width, height):
  # loop forever

  while True:
    # make up a random position

    x = random.randint(0, width-1)
    y = random.randint(0, height-1)

    # check if the character at the position is a space

    if scr.inch(y, x) == ord(" "):
      # if it is, replace it with a number and return

      scr.addch(y, x, ord("0") + random.randint(1, 9))

      return

# function to add an obstacle to an empty place on the screen

def add_block(scr, width, height):
  # loop forever

  while True:
    # make up a random position

    x = random.randint(1, width-2)
    y = random.randint(1, height-2)
```


GAME OVER
8

a (slightly) more complex program

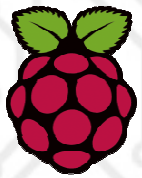
```
  I A  snake.py (Modified)(python)  import random       Row 5    Col 1    5:55  Ctrl-K H for help
import curses
import time
import sys
import random

# function to add a number to an empty place on the screen

def add_number(scr, width, height):
  # loop forever

  while True:
    # make up a random position

    x = random.randint(0, width-1)
    y = random.randint(0, height-1)

    # check if the character at the position is a space

    if scr.inch(y, x) == ord(" "):
      # if it is, replace it with a number and return

      scr.addch(y, x, ord("0") + random.randint(1, 9))

      return

# function to add an obstacle to an empty place on the screen

def add_block(scr, width, height):
  # loop forever

  while True:
    # make up a random position

    x = random.randint(1, width-2)
    y = random.randint(1, height-2)
```



- A series of examples, building up to a simple game of Snake, can be downloaded and unpacked by typing

wget http://www.raspberrypi.org/game.tar.gz

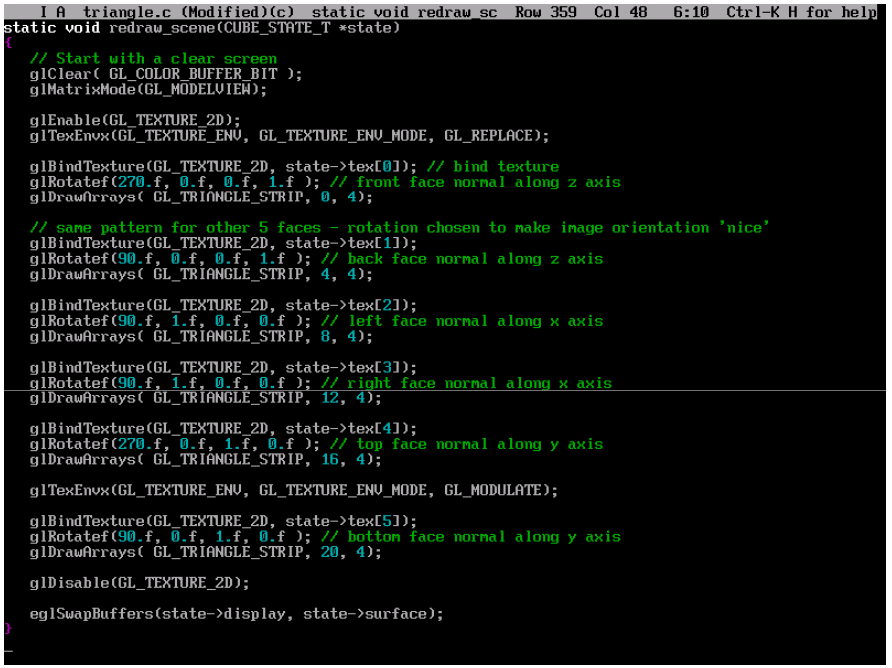
tar xvfz game.tar.gz

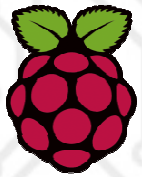


element14
www.element14.com

an OpenGL ES graphics program in C

- Raspberry Pi incorporates a powerful graphics accelerator
- We bundle a simple example
  - Written in C, using OpenGL ES
  - Source can be found in **/opt/vc/src/hello_pi/hello_triangle**
- To run the example
  - Change directory using **cd**
  - Build it using **make**
  - Run it by typing **./hello_triangle.bin**
- Try editing the source and the makefile using JOE



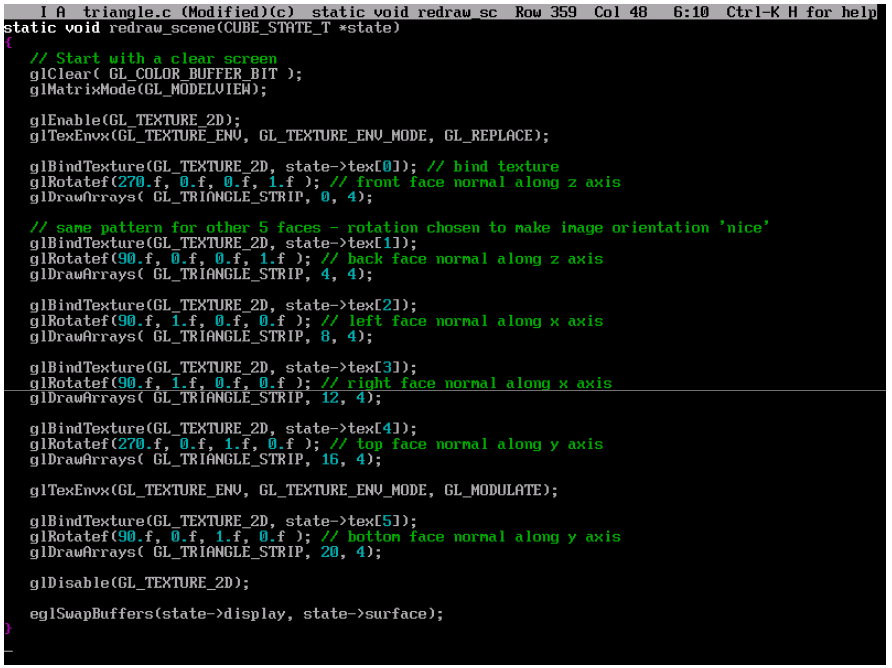


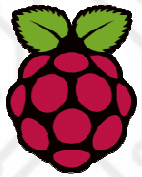
element14
www.element14.com

```
    I A  triangle.c (Modified)(c)  static void redraw_sc  Row 359  Col 48   6:10  Ctrl-K H for help
static void redraw_scene(CUBE_STATE_T *state)
{
   // Start with a clear screen
   glClear( GL_COLOR_BUFFER_BIT );
   glMatrixMode(GL_MODELVIEW);

   glEnable(GL_TEXTURE_2D);
   glTexEnvx(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);

   glBindTexture(GL_TEXTURE_2D, state->tex[0]); // bind texture
   glRotatef(270.f, 0.f, 0.f, 1.f ); // front face normal along z axis
   glDrawArrays( GL_TRIANGLE_STRIP, 0, 4);

   // same pattern for other 5 faces - rotation chosen to make image orientation 'nice'
   glBindTexture(GL_TEXTURE_2D, state->tex[1]);
   glRotatef(90.f, 0.f, 0.f, 1.f ); // back face normal along z axis
   glDrawArrays( GL_TRIANGLE_STRIP, 4, 4);

   glBindTexture(GL_TEXTURE_2D, state->tex[2]);
   glRotatef(90.f, 1.f, 0.f, 0.f ); // left face normal along x axis
   glDrawArrays( GL_TRIANGLE_STRIP, 8, 4);

   glBindTexture(GL_TEXTURE_2D, state->tex[3]);
   glRotatef(90.f, 1.f, 0.f, 0.f ); // right face normal along x axis
   glDrawArrays( GL_TRIANGLE_STRIP, 12, 4);

   glBindTexture(GL_TEXTURE_2D, state->tex[4]);
   glRotatef(270.f, 0.f, 1.f, 0.f ); // top face normal along y axis
   glDrawArrays( GL_TRIANGLE_STRIP, 16, 4);

   glTexEnvx(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);

   glBindTexture(GL_TEXTURE_2D, state->tex[5]);
   glRotatef(90.f, 0.f, 1.f, 0.f ); // bottom face normal along y axis
   glDrawArrays( GL_TRIANGLE_STRIP, 20, 4);

   glDisable(GL_TEXTURE_2D);

   eglSwapBuffers(state->display, state->surface);
}
_
```

an OpenGL ES graphics program in C

- Raspberry Pi incorporates a powerful graphics accelerator
- We bundle a simple example
  - Written in C, using OpenGL ES
  - Source can be found in **/opt/vc/src/hello_pi/hello_triangle**
- To run the example
  - Change directory using **cd**
  - Build it using **make**
  - Run it by typing **./hello_triangle.bin**
- Try editing the source and the makefile using JOE

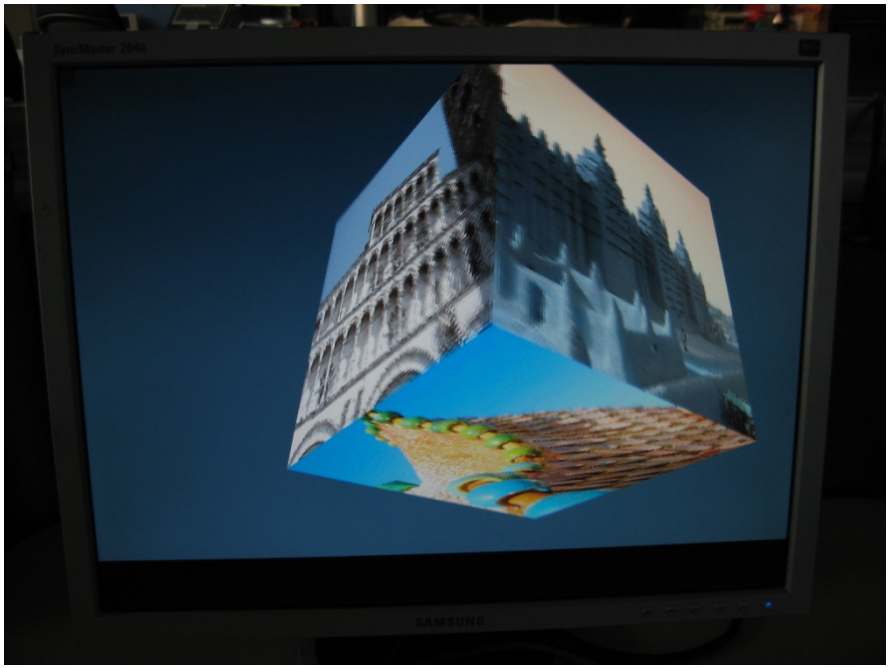




element14
www.element14.com

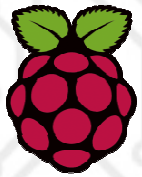
an OpenGL ES graphics program in C

```
pi@raspberrypi:~$ cd /opt/vc/src/hello_pi/hello_triangle/
pi@raspberrypi:/opt/vc/src/hello_pi/hello_triangle$ make
cc -DSTANDALONE -D__STDC_CONSTANT_MACROS -D__STDC_LIMIT_MACROS -DTARGET_POSIX -D_LINUX -fPIC -DPIC -
D_REENTRANT -D_LARGEFILE64_SOURCE -D_FILE_OFFSET_BITS=64 -U_FORTIFY_SOURCE -Wall -g -DHAVE_LIBOPENMA
X=2 -DOMX -DOMX_SKIP64BIT -ftree-vectorize -pipe -DUSE_EXTERNAL_OMX -DHAVE_LIBBCM_HOST -DUSE_EXTERNA
L_LIBBCM_HOST -DUSE_VCHIQ_ARM -Wno-psabi -I/opt/vc/include/ -I/opt/vc/include/ -I./ -I../libs -g -c
triangle.c -o triangle.o -Wno-deprecated-declarations
cc -o hello_triangle.bin -Wl,--whole-archive -L/opt/vc/lib/ -lWFC -lGLESv2 -lEGL -lopenmaxil -lbcm_h
ost ../libs/libilclient.a triangle.o -Wl,--no-whole-archive -rdynamic
rm triangle.o
pi@raspberrypi:/opt/vc/src/hello_pi/hello_triangle$ ./hello_triangle.bin
```

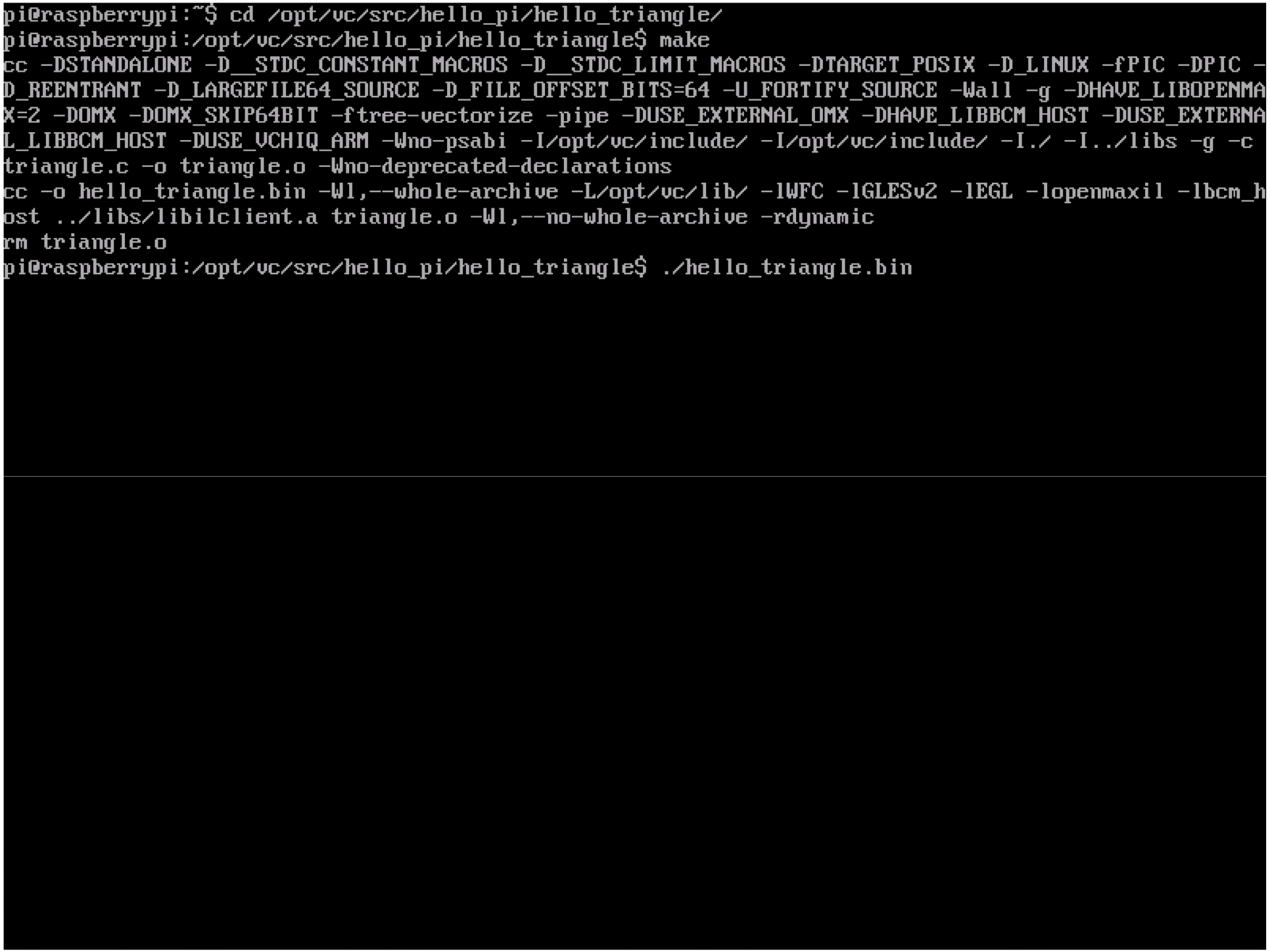


- More complicated examples available online, including Quake 3 at

https://github.com/raspberrypi/quake3



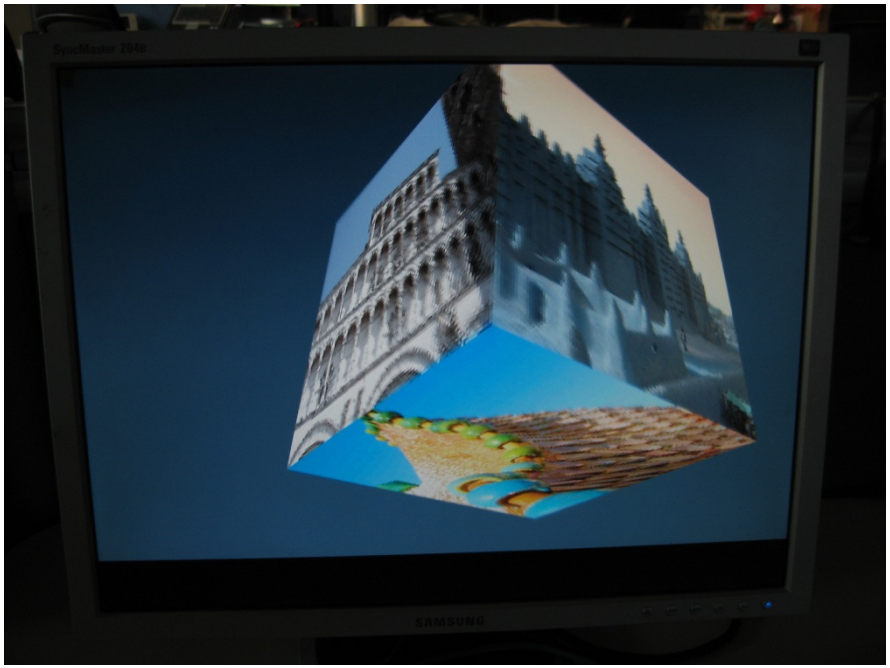
element14
www.element14.com


pi@raspberrypi:~$ cd /opt/vc/src/hello_pi/hello_triangle/
pi@raspberrypi:/opt/vc/src/hello_pi/hello_triangle$ make
cc -DSTANDALONE -D__STDC_CONSTANT_MACROS -D__STDC_LIMIT_MACROS -DTARGET_POSIX -D_LINUX -fPIC -DPIC -
D_REENTRANT -D_LARGEFILE64_SOURCE -D_FILE_OFFSET_BITS=64 -U_FORTIFY_SOURCE -Wall -g -DHAVE_LIBOPENMA
X=2 -DOMX -DOMX_SKIP64BIT -ftree-vectorize -pipe -DUSE_EXTERNAL_OMX -DHAVE_LIBBCM_HOST -DUSE_EXTERNA
L_LIBBCM_HOST -DUSE_VCHIQ_ARM -Wno-psabi -I/opt/vc/include/ -I/opt/vc/include/ -I./ -I../libs -g -c
triangle.c -o triangle.o -Wno-deprecated-declarations
cc -o hello_triangle.bin -Wl,--whole-archive -L/opt/vc/lib/ -lWFC -lGLESv2 -lEGL -lopenmaxil -lbcm_h
ost ../libs/libilclient.a triangle.o -Wl,--no-whole-archive -rdynamic
rm triangle.o
pi@raspberrypi:/opt/vc/src/hello_pi/hello_triangle$ ./hello_triangle.bin


SyncMaster 204B
SAMSUNG

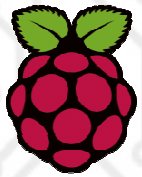
an OpenGL ES graphics program in C

```
pi@raspberrypi:~$ cd /opt/vc/src/hello_pi/hello_triangle/
pi@raspberrypi:/opt/vc/src/hello_pi/hello_triangle$ make
cc -DSTANDALONE -D__STDC_CONSTANT_MACROS -D__STDC_LIMIT_MACROS -DTARGET_POSIX -D_LINUX -fPIC -DPIC -
D_REENTRANT -D_LARGEFILE64_SOURCE -D_FILE_OFFSET_BITS=64 -U_FORTIFY_SOURCE -Wall -g -DHAVE_LIBOPENMA
X=2 -DOMX -DOMX_SKIP64BIT -ftree-vectorize -pipe -DUSE_EXTERNAL_OMX -DHAVE_LIBBCM_HOST -DUSE_EXTERNA
L_LIBBCM_HOST -DUSE_VCHIQ_ARM -Wno-psabi -I/opt/vc/include/ -I/opt/vc/include/ -I./ -I../libs -g -c
triangle.c -o triangle.o -Wno-deprecated-declarations
cc -o hello_triangle.bin -Wl,--whole-archive -L/opt/vc/lib/ -lWFC -lGLESv2 -lEGL -lopenmaxil -lbcm_h
ost ../libs/libilclient.a triangle.o -Wl,--no-whole-archive -rdynamic
rm triangle.o
pi@raspberrypi:/opt/vc/src/hello_pi/hello_triangle$ ./hello_triangle.bin
```



- More complicated examples available online, including Quake 3 at

https://github.com/raspberrypi/quake3



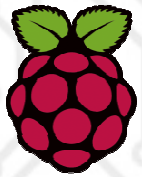
element14
www.element14.com

the configuration file (advanced users)

- At startup, Raspberry Pi reads **config.txt** from the SD card
  - Controls display and overclocking
  - Edit from a PC or on device using **joe /boot/config.txt**
- Common options include
  - arm_freq set ARM clock speed
  - gpu_freq set GPU clock speed
  - sdtv_mode select PAL/NTSC
  - hdmi_mode force HDMI resolution
  - overscan_* set screen border
- Very easy to break your install

- A typical configuration file

```
# select 16:9 PAL
sdtv_mode=2
sdtv_aspect=3

# medium size borders
overscan_left=28
overscan_right=28
overscan_top=16
overscan_bottom=16
```



wrap up

- We've seen how to
  - Set up, boot and configure your Raspberry Pi
  - Create and edit text files using the JOE editor
  - Run a simple Python script
  - Download and unpack more examples
  - Build and run one of the bundled C programs
- Remember Raspberry Pi is just a GNU/Linux box
  - Many books and online tutorials available
- Don't be afraid to play around with software
  - At worst you'll have to reflash your SD card