Raspberry Pi:

If you're eagerly waiting for the delivery of your Raspberry Pi, let **Jonathan Roberts** get you up to speed on this super diminutive device.

Our expert

Super-secretive **Jonathan Roberts** has been tinkering with Linux since he was a teenager. Which actually wasn't that long ago…

After a long wait, the eagerly anticipated Raspberry Pi has arrived. Here at **LXF** Towers, we've been lucky enough to be among the first to receive one of these bare-bones, developer preview boards, and we want to put it to good use over the coming months. In this tutorial, we're going to start by providing an introduction that will get any user, no matter what your past computing experience, up and running. We'll be covering:

- » What peripherals you need
- » How to create a bootable SD card
- » What to do when the system first starts
- » What settings you might want to consider tweaking
- » What you can, and can't, do with the RPi now

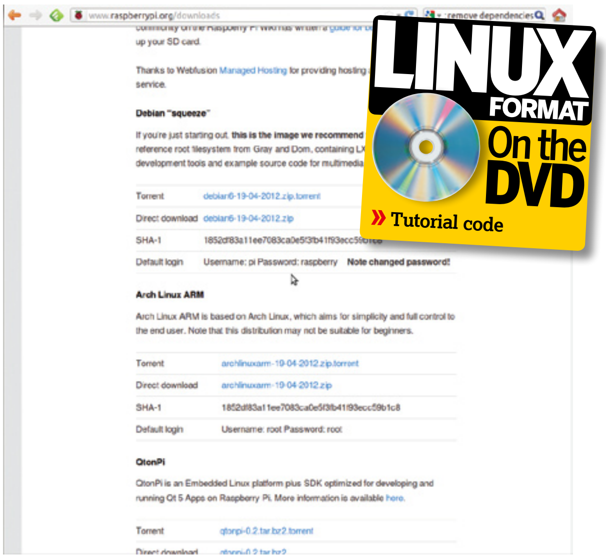
We hope that, whether you're waiting for your order to arrive or thinking about buying one later in the summer, this guide will show you what to expect and get you off to a flying start. Next month, we'll move on from getting everything working to begin putting the Raspberry Pi to its intended use – teaching you how computers work. We'll be looking at Scratch, one of the introductory programming languages included with it, to get your brain thinking more logically.

Peripherals needed!

Before ordering a Raspberry Pi, be aware that when it arrives it's just a green board with some electronics soldered on to it. That is to say, it's missing a number of vital peripherals, including:

- » USB keyboard and mouse
- » SD card for storage (and an SD reader)
- » Ethernet cable for network connection
- » Power supply
- » HDMI/Composite cable for display (and the display itself!)

This might sound like a nuisance, but meeting these extra requirements is simple most of the time. For starters, if you already own a desktop computer, you probably have all of these peripherals lying around the house. If you don't, then you can either buy one of the pre-compiled bundles when you order your Pi, or go to town and do a bit of shopping.



community on the Raspberry Pi Wiki has written a guide for be
up your SD card.

Thanks to Webfusion Managed Hosting for providing hosting
service.

Debian "squeeze"

If you're just starting out, **this is the image we recommend**
reference root filesystem from Gray and Dom, containing LX
development tools and example source code for multimedia

Torrent	debian6-19-04-2012.zip.torrent
Direct download	debian6-19-04-2012.zip
SHA-1	1852df83a11ee7083ca0e5f3fb41f93ecc59b1c8
Default login	Username: pi Password: raspberry **Note changed password!**

Arch Linux ARM

Arch Linux ARM is based on Arch Linux, which aims for simplicity and full control to the end user. Note that this distribution may not be suitable for beginners.

Torrent	archlinuxarm-19-04-2012.zip.torrent
Direct download	archlinuxarm-19-04-2012.zip
SHA-1	1852df83a11ee7083ca0e5f3fb41f93ecc59b1c8
Default login	Username: root Password: root

QtonPi

QtonPi is an Embedded Linux platform plus SDK optimized for developing and running Qt 5 Apps on Raspberry Pi. More information is available here.

Torrent	qtonpi-0.2.tar.bz2.torrent
Direct download	qtonpi-0.2.tar.bz2

› **All of the operating systems that are compatible with the Raspberry Pi are downloadable from their website.**

Before you open your wallet, however, read these notes to make sure that you buy components compatible with the RPi.

» **The power supply** has to provide 5V/1A output, and have a mini-USB connection. Because of a new EU law, many mobile phone chargers match these specifications, so check if yours does. Here at **LXF** Towers, we've been using our Android mobile phone charger.

» **The SD card** must be Class 4 and have at least 4GB of space. Of course, the more space, the more applications and data you'll be able to keep on your Raspberry Pi. If you're unsure, there's a list of SD cards known to work at **http://bit.ly/wVO5By**. You'll need to be able to read and write to this SD card from a computer other than the Raspberry Pi, so if you don't have a reader built in, you'll need to buy a USB one.

» **The HDMI/Composite cable** doesn't need to be expensive! Some shops sell very expensive ones, but you can

Alternative card preparations

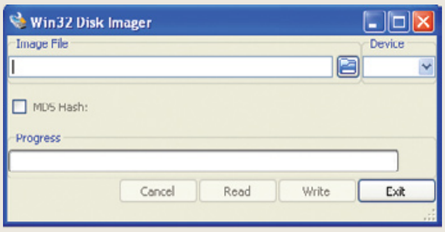
If you don't have a machine running Linux, you won't have access to **dd**, the tool we used to copy the operating system to the Raspberry Pi's SD Card.

You can still copy the operating system to the SD card, however, using a program called *Win32DiskImager*.

It was originally developed to work with the Netbook Remix of Ubuntu, but it has now become a popular tool for people playing with Arm devices (such as the Raspberry Pi). You can download it from **http://bit.ly/Dh2nE.** It's a simple tool. Select the .img file in the first field, and select the SD card from the Device field next to that. Then, all you need to do is click Write.

Make sure you get the correct drive, and consider backing up all your data first, because it's possible to lose everything if you make a mistake.

Also, as with **dd**, this tool will erase all information on the SD card.



› **This tool can put the OS on the SD card (image from the Ubuntu wiki, CC-BY-SA 3.0).**

getting started

buy them from Poundland or Tesco for about £4, and they will work perfectly. As for which you need, HDMI or Composite, well, that depends on what kind of screen you have.

Have a look at the image on the right. At the top is a HDMI connection, in the middle is a DVI connection, and below that is a Composite connection. Now look at your screen, whether it's a TV or a computer monitor, and see which it has – you'll then know which kind of lead to buy. If it has a DVI connection, buy a HDMI lead and a HDMI to DVI converter; if it has a Composite and HDMI connection, you should opt for the HDMI connection.

Operating system primer

As with the missing peripherals, the Raspberry Pi is also missing an operating system. To make it work, you need to get a compatible operating system and put it on the SD card using a special technique. Of course, if you don't know what an operating system is, this might be tricky, so let's start with that.

An operating system is the software that lets you interact with the computer: it receives your inputs, controls the hardware and sends output to the screen or over the network. Ubuntu, Windows and Mac OS X are all examples of operating systems. Without an operating system, a computer can't do anything – it's just a big piece of inert metal and silicon.

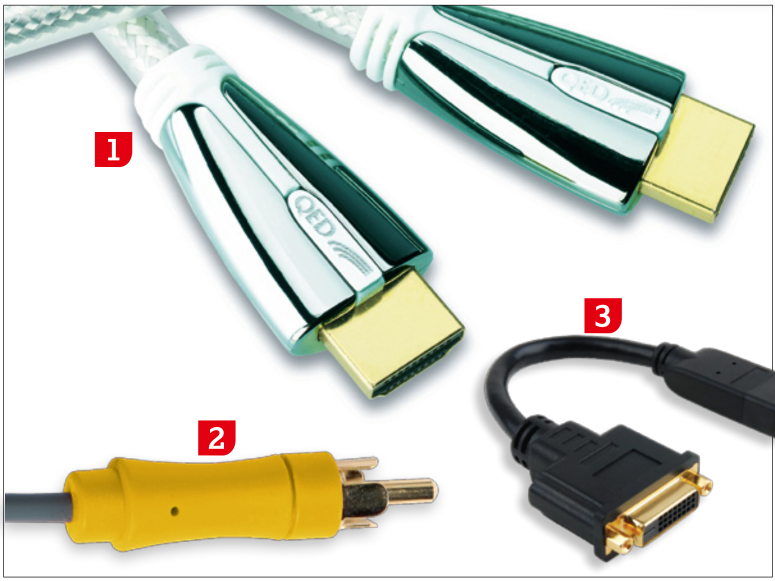
On a traditional computer, the operating system is stored on the hard drive. Every time you turn it on, the first thing it does is find the operating system and load it in to the memory (RAM). The operating system then takes over control of all the hardware and keeps everything working until you shut it down. It's often said that the computer 'runs from' or 'boots off' the hard drive. The Raspberry Pi, however, is not a traditional computer and doesn't have a hard drive. Instead, it uses the SD card to store the operating system, all the other software applications you install and all your personal data files. It runs off the SD card.

Getting the OS

To get the operating system on the SD card, you first need to download one. You can find all the compatible operating systems on the Raspberry Pi website: **www.raspberrypi.org/downloads**. They recommend that you download the Debian version for the time being, and this is what we're going to be using in this tutorial.

The download is quite large at 400MB+, so it will take a while. If you have a data cap on your network connection, you can also find the download on this month's coverdisc. Once you've got the image, you need to put the SD card in to the computer. If it has a built-in card reader, use that, but if it doesn't, you'll need to connect a USB one.

Now, the download is a ZIP file, which you should extract. Inside the newly-created folder will be a .img file – it's this that needs to be copied on to the SD card, but it can't be copied like a normal file. The .img file, as the name suggests, is an exact image of a device. It contains information about the layout of data on the device and files used to boot the computer, that have to be in exactly the right part of the drive. If you just copy it, the files won't be put in the right place, but will be stored on top of the disk's existing structures.



› Buying the right cable for your monitor or TV isn't tricky, you just need to look first. 1 HDMI cable, 2 Composite cable, and 3 HDMI to DVI converter.

Preparing the SD card

Instead, we need to use a special program that will copy the first byte of the image to the first byte of the disk, the second byte to the second byte, etc. On Linux, we use a command line tool called **dd** to do this.

Put the SD card in the computer, and then open a terminal window and type **df -H**. This will display a list of all storage devices and partitions on your computer, along with their size, allowing you to find out what device label your computer has assigned to the memory card.

Look closely at each line and find an entry that has the same size as your memory card and has Media under the mounted column on the right. You may not find one with exactly the same size as is stated on your memory card, but it will be close. Make a note of the information under the Filesystem column on this line, which should be something like **/dev/mmcblk0**.

Now, still on the command line, you need to enter the following (but read the following notes before pressing return!): **sudo dd if=/home/jon/Downloads/debian6-19-04-2012/debian6-19-04-2012.img of=/dev/mmcblk0**. Here, the part that follows **if** should be the path to the downloaded .img file. In this case, it's in the Downloads

»

» folder of my home directory. If you saved it elsewhere on your computer, you'll need to enter the full path to that location instead. The part after the **of** should be the part you noted down from the Filesystem column. If what you noted down looked like **/dev/mmcblk0p1**, you need to remove the **p1** from the end; if it looked like **/dev/sdc1**, you need to remove the number from the end.

Be aware that this will remove all data on the device, so make sure you back everything up first, and make sure that you get the right device! If you accidentally direct **dd** at your main hard drive, you're going to have serious problems.

Booting up

That's the only dangerous part done. Now, place the SD card in to the slot on the bottom of the Raspberry Pi, connect your USB keyboard, mouse and the display and Ethernet cables. Then, with the monitor turned on, connect the power lead, and the Raspberry Pi should boot itself.

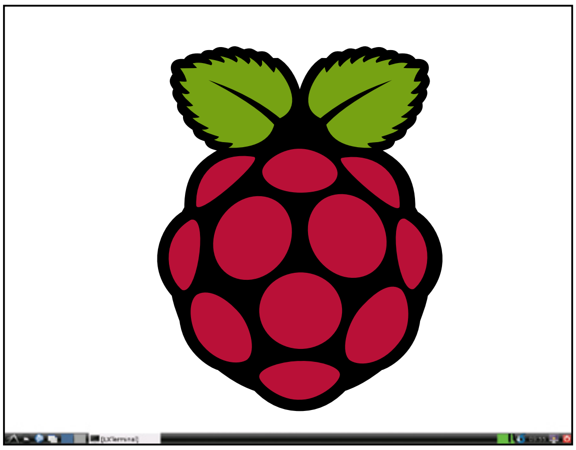
You'll see a red power light next to the USB connections, and shortly after you should see some flashing green lights appear. If you don't see these, it means something went wrong with the SD card setup. First, double check that it's pushed in to the slot properly, and then reconnect the power; if it still doesn't work, revisit the previous steps and try to configure the SD card again.

If all is well, however, you should see a lot of text scroll by on the screen with a Raspberry Pi logo at the top. The first time you start the Raspberry Pi, it will enter a one-off initialisation process. Along the way, it may stop loading when it says 'Stopping portmap daemon'; if it does stop at this point, you can simply turn the power off and on again, and it will start without trouble the next time around.

The prompt

Eventually, text will stop scrolling across the screen, and there will be a line at the bottom that reads '**raspberrypi login:**' followed by a blinking cursor. This is the login prompt, and it means the machine is waiting for you to enter your username and password. On the Debian image, the username is pi and the password raspberry. Type the username and press Return, and then type the password and press Return. It won't show your password as you type it, just in case anyone is looking over your shoulder. After doing that, you'll find the text stops at another line, similar to the last. This time it reads '**pi@raspberrypi:~$**' followed by a blinking line. This is the command prompt. The blinking line means that the prompt is ready to accept your commands, which you enter by typing in the names of different programs and pressing Return.

If you plan to use your Raspberry Pi to learn about computers and how they work, to run your own server or to do some programming, you'll spend more time with the command prompt, but for now let's get the graphical interface working.



› **The default Raspberry Pi desktop is based on LXDE, a lightweight interface that's ideal for systems with slower processors and less memory.**

Graphical caveats

The Raspberry Pi is unfinished. The boards that have been sent are developer previews, with the Foundation sending them in the hope that geeks will start tinkering, find and report problems, as well as creating lots of fun applications for when they start delivering refined versions to the target audience: school children.

With that in mind, you may be less surprised to learn that there are some problems with hardware support in the distributions. None of them have yet got *X*, the software that controls the display, working with the integrated (and very powerful) graphics system, which means that it's slow to redraw windows.

It's also impossible to get most existing 3D games to work. This is because most of the 3D games on Linux have been written to interact with the graphics system through something called OpenGL; the Raspberry Pi supports only a sub-set of this, however, called OpenGLES.

As time progresses, expect to see new games appear and old games modified to work with OpenGLES; and expect to see *X* gain support for the integrated graphics. Eventually, everything will work snappily – patience is all that's required.

Pretty pictures

At the prompt, type **startx** and press Return. The screen will flicker for a few moments, then turn black, then a huge Raspberry Pi logo will appear with a task bar across the button that's similar to Windows. This is LXDE, the Lightweight X11 Desktop Environment. On Linux, and unlike MacOS X or Windows, users can choose from many graphical interfaces to interact with their computers. LXDE isn't the prettiest or most modern of these, but it's one of the leanest. This makes it perfect for running on hardware without much processing power or memory, just like the Raspberry Pi.

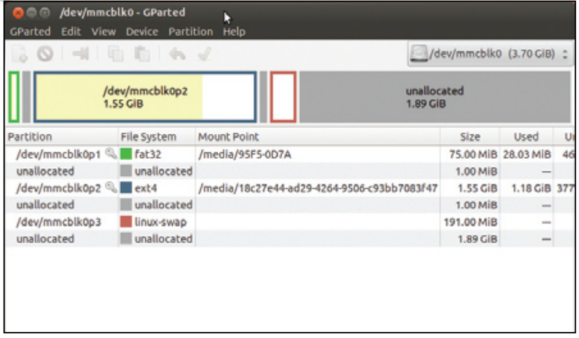
Have a play around and click the different buttons to see what they do – you can't break anything at this point. And even if you do, you can just unplug and reconfigure your SD card and you'll be back where you started.

Although you can't see the command line anymore, it's still running in the background. You can access it again by pressing Ctrl+Alt+F1 – try it. You should see some obscure looking text following the **startx** command you issued. This is the output from the running program, the *X Server*.

X is the program that Linux uses to control input devices and display information on the screen. It creates a series of virtual consoles, accessed by typing Ctrl+Alt+F1+F6. The graphical environment, on the Raspberry Pi at least, will always be accessible on the console, represented by F2. So, to get back to LXDE you can just type Ctrl+Alt+F2.

One thing you might notice as you play and explore is that the desktop doesn't fit your display properly. This is a common problem that people using HDMI connections

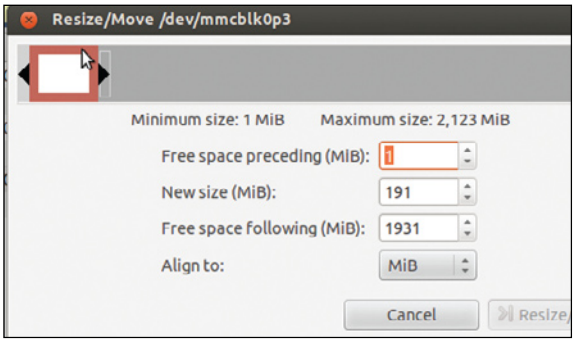
Step-by-step: Take back the space



/dev/mmcblk0 - GParted

/dev/mmcblk0 (3.70 GiB)

Partition	File System	Mount Point	Size	Used	U
/dev/mmcblk0p1	🟩 fat32	/media/95F5-0D7A	75.00 MiB	28.03 MiB	46
unallocated	unallocated		1.00 MiB	—	
/dev/mmcblk0p2	ext4	/media/18c27e44-ad29-4264-9506-c93bb7083f47	1.55 GiB	1.18 GiB	377
unallocated	unallocated		1.00 MiB	—	
/dev/mmcblk0p3	🟥 linux-swap		191.00 MiB	—	
unallocated	unallocated		1.89 GiB	—	



Resize/Move /dev/mmcblk0p3

Minimum size: 1 MiB Maximum size: 2,123 MiB

Free space preceding (MiB): 1

New size (MiB): 191

Free space following (MiB): 1931

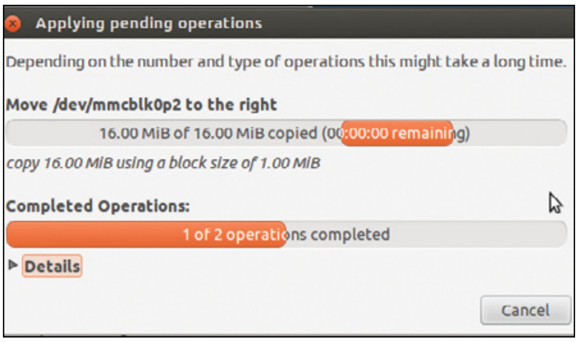
Align to: MiB

1 Pick a card

Select the right device! Look in the drop-down menu and choose a device that's the same size as your SD card. If it's the right one, you'll see three partitions listed in the main window, and two bits of unallocated space.

2 Move it around

Right-click the **linux-swap** partition and select Resize/Move. In the window that appears, drag the small box all the way to the right of the box that it's contained within.
Click Resize/Move.



Applying pending operations

Depending on the number and type of operations this might take a long time.

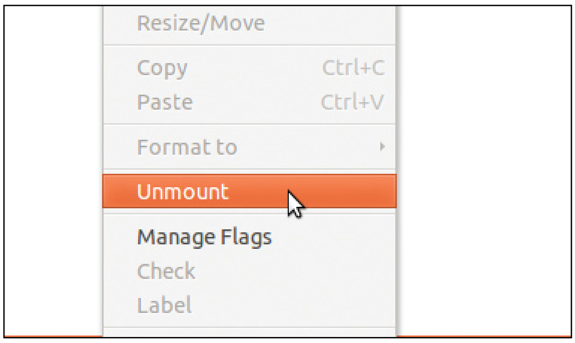
Move /dev/mmcblk0p2 to the right

16.00 MiB of 16.00 MiB copied (00:00:00 remaining)

copy 16.00 MiB using a block size of 1.00 MiB

Completed Operations:

1 of 2 operations completed



3 Make some room

Right-click the **ext4** partition, select Unmount, then right-click and select Resize/Move. Drag its right-hand edge to the end of the box to resize it. Click Resize/Move.

4 All done

Then click the green tick at the top of the screen. Wait patiently while *GParted* moves and resizes your partitions. Take the SD card out and reboot your Pi.

have found. It's caused by something called overscan, which is a hangover from televsion sets from the 1930s through to the 1990s.

Fixing the problem

To fix it, we're going to go back to the command line. We won't switch virtual consoles as we did before, but we'll use a 'terminal emulator'. Open the LXDE menu by clicking the icon in the bottom-left of the screen, then click Accessories > LXTerminal. This is another way to access the command line. Into the prompt which appears in a new window, you need to type **sudo leafpad /boot/config.txt**. This will launch another new window – a text editor called *Leafpad*. You don't need to open a file, since the command you entered also specified which file *Leafpad* should be opened to edit, **config.txt** in the **/boot** directory. Into here, type **disable_overscan=1** and then save the file. You can then reboot your Raspberry Pi by clicking the logo in the bottom-right of the screen, and when it comes back on the screen should be filled.

There are a lot more options you can put in this file, and they're all documented on the Raspberry Pi wiki: **http://elinux.org/RPi_config.txt**. A lot of these have to do with display modes, including resolution, etc, but some also cover modifying how fast the processor runs – overclocking it. You should avoid these, as unless you know what you're doing, they can damage the computer. The hdmi_drive option is useful if you're using a HDMI to DVI converter. Set it to 1 to ensure your screen is at the correct resolution and reboot.

At this point, you've got your Raspberry Pi running, you know how to start the graphical interface and you've set the screen resolution. There's one more tweak we want to describe, and that's how to configure your SD card to make all its storage space avaiable for files and software. The reason this is necessary is when you put the .img file on to the SD card, it automatically divided the drive into three parts. Each of these appears to the computer as if it were another physical hard drive, and is known as a partition. One of these is used for storing the operating system and all your files (known as the root parition, represented by a **/**), another is used for providing virtual memory (known as the swap space) and the third is used for storing the files that let the Raspberry Pi start. Because each is seen as another physical device, if there's room on the SD card that's outside of these partitions, the operating system will ignore it.

This is where the trouble lies. The Debian image we put on the SD card creates a root partition that's always the same size. As a result, on our 4GB SD card we found ourselves with 2GB of unused space, and only a few hundred megabytes to store new applications and files. To fix this, you'll need a Linux machine with *GParted* installed on it. You can then follow our step-by-step guide. Remember to back up your data first. LXF