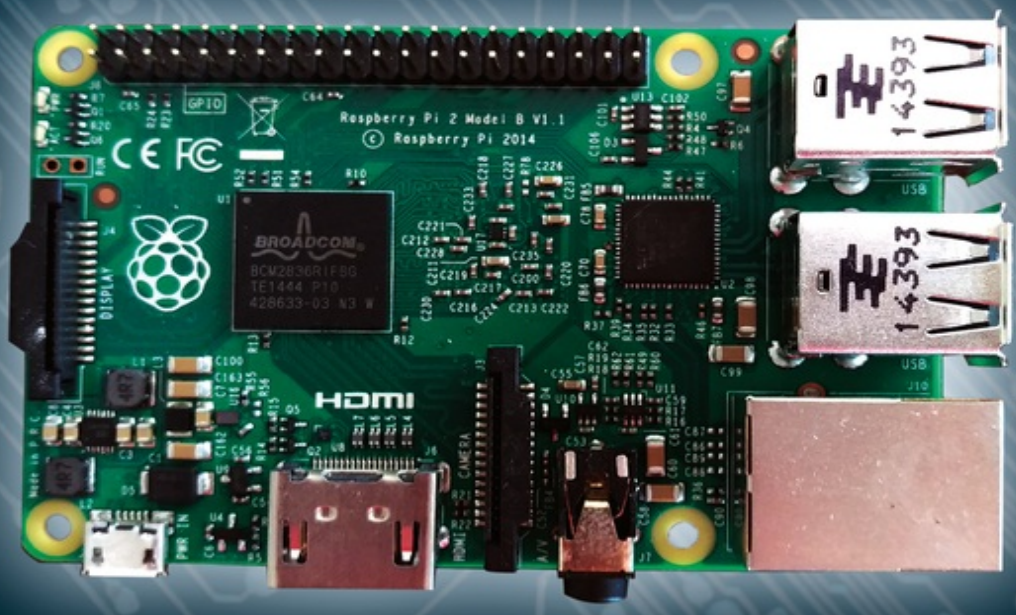
TECHNOLOGY IN ACTION™

SECOND EDITION

Learn Raspberry Pi 2 with Linux and Windows 10

LEARN THE INS AND OUTS OF LINUX AND HOW TO INSTALL WINDOWS 10 ON THE RASPBERRY PI 2



Peter Membrey and David Hows

Learn Raspberry Pi 2 with Linux and Windows 10

Second Edition

Peter Membrey

David Hows

Any source code or other supplementary material referenced by the author in this text is available to readers at www.apress.com/9781484211632 . For additional information about how to locate and download your book's source code, go to www.apress.com/source-code/ . Readers can also access source code at SpringerLink in the Supplementary Material section for each chapter.

ISBN 978-1-4842-1163-2 e-ISBN 978-1-4842-1162-5
DOI 10.1007/978-1-4842-1162-5

Learn Raspberry Pi 2 with Linux and Windows 10

Managing Director: Welmoed Spahr
Lead Editor: Michelle Lowman
Technical Reviewer: Brendan Horan
Editorial Board: Steve Anglin, Louise Corrigan, Jonathan Gennick, Robert Hutchinson, Michelle Lowman, James Markham, Susan McDermott, Matthew Moodie, Jeffrey Pepper, Douglas Pundick, Ben Renow-Clarke, Gwenan Spearing, Steve Weiss
Coordinating Editors: Kevin Walter and Mark Powers
Compositor: SPi Global
Indexer: SPi Global
Artist: SPi Global
For information on translations, please e-mail rights@apress.com, or visit www.apress.com .
Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotiona use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales–eBook Licensing web page at www.apress.com/bulk-sales .

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC an the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a Delaware corporation.

For ~~Dr. John McDougall, Mary McDougall, Dr Caldwell Esselsytn, Dr T Colin Campbell and Dr~~ Doug Lisle, for doing the right thing against all the odds.

Peter Membrey

To my parents, Peter and Bev, for all their support.

David Hows

Introduction

Despite sounding like something grandma would bake on Sunday afternoons or a noise that would make people glare and tut, the Raspberry Pi is in fact a computer. That much you probably knew (although let's be honest the name and logo don't really give much away) but actually, the Raspberry Pi promises more than that. An awful lot more.

The venerable Commodore 64 was released in 1982 and with sales reaching upwards of 17 millio it is often considered the best selling computer of all time. More importantly (at least from my perspective) it was also my first computer. For Christmas, just before my ninth birthday (so when the C64 was nearly a decade old) I received the new model (C64C) which was identical to the classic machine in all but cosmetics. It arrived all set up and attached to a nice new 14" television (it even ha a remote control!). I suspect my dad had hatched what he believed to be a most cunning plan; if he could sneak in and set everything up whilst I was asleep, then come Christmas morning, I would be s busy playing with the computer that my parents might get an extra few minutes sleep.

Sadly things did not go quite according to plan. Although everything was set up and even though the television was tuned to the computer's signal, one simple but key thing had been forgotten. It hadn't occurred to anyone to tell me how to actually load a game. Needless to say a lie in was not forthcoming…

Games came on cassette tape (this was before CDs - what do you mean what's a CD?) and at leas on the C64 had to be played in a special tape recorder called a datasette. Sadly the datasette spent more time in the shop than being attached to my computer and as it was the only way to load anythin into it, I had no choice but to occupy myself with the manual. This I used to great effect and taught myself how to program good old BASIC (Beginners All Purpose Symbolic Instruction Code - can yo believe I actually remembered that?).

While I'm sure this story is very gripping, you could be forgiven for wondering why I am boring you with it. Those events happened over two decades ago (boy does time fly) so what possible relevance could they have today? Well tinkering with that machine then and then the Amiga that followed it (still my favorite machine of all time) gave me a real appreciation for what a computer could do. The Amiga was essentially severely under powered compared to PCs of the same era and y it consistently beat them with better graphics, better reliability and better sound. It was able to do all of this because the hardware was exquisitely designed. Amiga enthusiasts were some of the most resourceful people I've ever seen. Who'd have thought you could turn a real time clock port into a connector for high speed storage?

All of this was only possible because people really understood how all the parts fitted together. They knew how to get the best out of the machine because they really new how the machine worked. These days I spend my working day trying to make fast things go faster. To stand any hope of succes I too need to know how everything works. Companies need people like me to push things forward bu they're coming across a bit of a problem. People who really know computers inside out are getting much harder to find - we are a dying breed and this is the situation that the Raspberry Pi Foundation desperately trying to reverse.

So what happened? Well things changed. Computers went from being the curiosity in the corner t being a basic part of everyone's lives. They evolved to the point where for the most part they just work and everyone knows how to use them. This is similar to the family car. Everyone has a rough idea how a car works, but few people are very interested. The car takes them from place to place and that ability is what is interesting, not how the car achieves it. Computers are generally seen in the same light. People have a rough idea about turning them on, installing software and so forth, but how

they actually work at a low level isn't really seen as relevant or interesting. This in turn means that n only are fewer people getting excited by computing itself, but even less people think that there's mor to it than double clicking an icon.

This problem has drifted up through schools and into universities. Teaching programming is a relatively challenging task. It requires a certain way of thinking that for many people is tough to get a handle on. Traditionally universities would start a computer science course by teaching about logic gates, how memory works and how to program a CPU. Only once you understood what the bare meta was doing would you try to learn C because although C is a higher level language, it reflects the hardware it runs on. Understanding the hardware makes understanding C that much easier.

But with larger class sizes, more limited teaching time and students arriving with less and less knowledge of computing fundamentals, universities have had to adapt. Rather than teaching all that low level stuff, now they teach Java and scripting languages such as Python. Because these languages handle all of the fiddly bits for you, you can effectively pretend that they don't exist (although this ca cause some issues, see "The infinite memory myth"). This is simply fantastic from a productivity point of view, but when you do want to take it to the next level (maybe you're processing data and your script is just too slow) you have no idea where to turn. Worse, when someone tells you the technique for improving that performance, you have no idea what they're talking about.

Of course not all universities have taken this route. I'm studying at the Hong Kong Polytechnic University and their course on Computer Architecture is very detailed and covers a lot of ground. If you want to get the top grade you will need to implement a CPU cache for the CPU simulator program. Needless to say, there is a lot to learn for everyone on this course. That said, we need more than this. It's too late to capture people's interest when they're starting graduate studies. I taught 7 year olds how to program BBC BASIC when I was in my last year of primary school (they even got presented certificates by the school) and they loved it. Computing lets you create a virtual world with your mind (they liked to create little text based adventure games) and ultimate power rested in their hands. They got creative, they added spells, new roads, secret entrances and much more. Okay, they needed a helping hand (they were only 7) but they had the desire to create and to build cool new things.

INFINITE MEMORY MYTH

Over the years I've done a lot of consultancy work with large enterprise customers and that has inevitably meant I've come across Java on many occasions. One of the interesting things I have come across is what I've termed the Infinite Memory Myth. This seems to crop up more in Java applications than in other languages, but that's probably because Java tends to be more widely used in those settings.

The short version of the myth is that developers seem to constantly create new objects, often to the point where the application consumes huge amounts of memory or crashes altogether. They tend to have no idea how much memory each object takes or more worryingly why they should care. As far as they are concerned, they request a new object and one is provided. When an object is no longer used (i.e. nothing points to it any more) Java will at some point get around to cleaning it up (called garbage collection). All of this is automatic; the developer doesn't need to do anything.

The problem is this leads people to forget (or in many cases were never taught at all) that memory is finite and at some point it simply runs out. You can't assume that you can read in every row in a table and that it will always work. You can't assume that just because your test file is 50MB in size that the application will never be given a 5,000MB file to work on.

This lack of understanding stems from not being able to appreciate all the hard work Java is doing on the programmer's behalf. It is running about and managing memory allocation and

garbage collection, and the programmer remains blissfully unaware. A good understanding of computing fundamentals would give a developer keen insight into what Java is doing (both the how and the why) and thus appreciate that just because creating new objects is easy, memory itself is not free.

So this is what Eben Upton and the Raspberry Pi Foundation are trying to bring back to the world. They want to rekindle that lost art and make computers cool and interesting again. To do that they have created a computer that even by today's standards is no slouch. Is it as powerful as your laptop? Well no, probably not but can you buy your laptop for $35, slip it into your pocket (possibly not a great idea), generate relatively little heat and drink very little in the way of power? If you answered yes to those questions, I really really want to hear from you - that sounds like a laptop I need to buy!

However you choose to look at it the main selling point (no pun intended) is the price. Anyone ca pick up a Raspberry Pi without having to think too much about it. With a modest laptop clocking in a around $500 + and a MacBook pro nearly four times that, it's not the sort of thing you can just splurg on without thinking, especially if it's going to be for experimentation and playing about. However at $35, the Pi is cheaper than some monthly movie subscriptions, you could almost buy a new Pi every month!

Why eat Raspberry Pi?

Whichever way you look at it, you will come back to the price. Whatever else the Raspberry Pi is and whatever other promises it has in store for us, all of them are interesting because of the price. There are two types of people who will be rushing out to get a Pi. The first group are already clued up on Linux and computing and for them the Pi represents a server in a pocket and a cheap one at that. No longer do they need a full size PC monster guzzling electricity and generating enough heat to rival a bar heater. Oh sure you can get low powered systems in nice shiny boxes, but they're still not all that cheap to buy even if they're cheaper to run. However a device like the Pi is cheap to run and cheap to buy and it has just what you need to build a pretty respectful server.

If you're not one of those people then don't worry, because this book is for you. You like the idea of a cool little computer for $35 and you think you can do some pretty awesome stuff with it; you're just not really sure how. For us the big benefit is that the Pi is at the sort of price where we can afford to buy it just for fun and use it for experimentation. It's not going to replace the family PC and you're not going to need to take out a mortgage to buy one. You can play around with a Pi completely guilt free and try all manner of weird and interesting things without having to worry about cost or destroying your main computer (and thus incurring the wrath of everyone you live with).

Because the Pi is close enough to a normal PC (even though the architecture is a bit different) you can do PC type things with it. In fact, that's the first thing we show you in this book. Thus you don't have to start from scratch, all that you know already you can apply to a Pi (it rhymes if you say that really fast) and so you can hit the ground running. No doubt you will want to do all the things that the first group of people want to do as well. Fear not for we have you covered – by the time you've finished this book you'll be able to do all that and more!

There are lots of reasons why everyone should rush out right now and get some Pi. Actually at the time of writing, there is still a lead time for delivery and when the Pi was initially released, one of the resellers took 100,000 pre-orders in just one day. That's a lot of Pi! Although the lead time will naturally keep changing, the short version is that the sooner you order, the sooner you will get your P

So why all of this interest? What is so special about the Raspberry Pi that it has achieved an almost cult-like following and is still pretty hard to get your hands on?

It only costs $25

Okay, hands up all of you who are only interested in the Pi because it costs significantly less than a night on the town? If you put up your hand, you're not alone. The goals of the Raspberry Pi Foundation are laudable but they all center around getting this powerful machine into our mitts at a price point that won't break the bank.

What really has everyone drooling is not the fact that as far as computer hardware goes, the Pi is effectively free, but more that it is a full computer that can run Linux. That means servers, home automation, video streaming and pretty much anything else you can imagine.

WAIT, IS IT $25 OR $35?

Throughout the book we do bring up the price of the Pi a fair bit, after all it is one of its most distinctive features. However we also mention two prices, $25 and $35, so which is it? Well there are two versions of the Pi, the Model A and the Model B (as uninspiring as those names sound, they're taken from the BBC computers and from a geeky point of view, the names are quite inspired). There isn't a great deal of difference between the two models, with the Model B having 10/100 Ethernet built in and an additional USB port. The Model B also draws a lot more power. The Ethernet adapter is actually connected internally via USB so there is no difference between the built in Ethernet and a USB device that you could plug into the USB port itself.

So which should you buy? Well if you think networking will be useful, the built-in Ethernet port is pretty much required. I love having built-in Ethernet connectivity as it just makes life so much easier. However if you aren't planning on using it all that much, then there really is no need to get the Model B. That said, for $10, it might be worth splashing out just in case you decide you want to play with networking later on…

MOAR PI!

If one Pi is good, then two must be better right? Well, $70 would get you two Pis and while this doubles the cost, it also more than doubles the fun. Now you can experiment with networking and getting the Pis to talk to each other. After all, it's good to talk!

Experiment in safety

I don't know about you but when my computer is out of commission for even a short period of time i is pretty inconvenient. I certainly don't want to fiddle about with something and accidentally erase m hard disk (been there, done that). You'd also be well advised not to try over clocking your CPU on th brand new computer that you were just given as it's not much fun to think you've totaled the machin within an hour of turning it on!

To kill hardware takes a fair bit of effort (such as taping over a pin on the CPU and removing the CPU speed multiplier lock hehe) but it's fairly easy to remove your family photo album and your latest draft of the book you've been working on for the past six months (there is a reason why we dedicated a whole chapter and personal plea in Practical Guide to Performance to backups). If you have a Pi and you nuke it, the worst case is $35 down the drain which is a lot better than what would happen if you toasted your main machine.

Independence

I'm sure some people will point out (quite correctly) that most of the horrors I just described can be avoided if you play in a VM rather than on your main machine. Apart from that not being as fun (rea hardware just smells better) it doesn't give you all the benefits of a separate piece of kit. For a start,

virtual machine is only on when your main machine is running. If you happen to have a laptop which follows you everywhere, a virtual machine won't be a great option for a home web server. Also if you ever reboot your main machine, your virtual machine will go down with it. If you were using it to stream movies to your TV, you could end up with some very displeased family members. By having a real piece of hardware you can keep your experimentation completely separate from anything else that you might be doing.

Low Power

The Pi has a very modest power footprint. In fact, the Model A Pi only draws 300ma, which means you can power the whole thing from your USB port. According to Apple, my iMac draws 94 watts at idle and up to 241 watts when the CPU is maxed out. The Model A Pi however draws at most 1.4 watts and the Model B draws at most 3.5. That's an awful lot of power saved. Understandably, these figures only take into account the power requirements of the Pi itself with a bit extra put aside to power modest USB devices. If you add lots of power hungry devices to your Pi, these figures would increase accordingly.

APPLES TO ORANGES

I just know someone is going to cry foul about my power comparison and for good reason. It is true that compared to an iMac, the Pi draws basically no power at all. However, it is also true that an iMac does a lot more. First off it has a big screen, hard disk and a CPU that would blow the ARM on the Pi into the middle of next week. So if I know that I'm comparing two totally different systems and I'm admitting that they're very different(who compares a golf cart to a Formula One car?) why am I wasting ink and paper with this description?

Well, although we are comparing two very different machines, we are comparing the same sort of tasks. If you want to have a little web server or stream video to your TV set, a Pi is more than powerful enough to do that for you. Bear in mind that the Pi is clocked in at 900Mhz and that not too long ago, that was what you'd find on a very powerful desktop. It's not that long ago that you'd find this sort of performance on an enterprise grade server. Ten years ago, it was just a fantasy. In short, the Pi has more than enough juice to do most of the things you'd want from a server and it won't require a small nuclear power station to do it.

The ingredients for a Raspberry Pi

After extolling all of its benefits you might be wondering why a $35 computer stacks up so well to ones that cost many times that. If you're thinking that, then it won't be long before you wonder why there is such a big difference in price. Surely if you could get something this cheap that does most of the things your main machine can do, then something must be up with the price of the other machine. After all, if a powerful laptop could be made significantly cheaper, it would easily make more money in increased sales than it would lose in reduced profit margin.

Well that's true to a point, but the Pi, powerful as it is, will probably never be a direct replacement for your main computer. It's not any particular one thing that limits the Pi, but a combination of design decisions to balance features with cost that will ultimately prevent it from taking the crown. That said, it is still a fantastic platform and we'll look at some of those highlights right now.

ARM CPU

The major and most obvious difference between the Pi and your desktop is that your desktop will

almost certainly have a processor from either Intel or AMD at its core whereas the Pi has an ARM based CPU. The CPU (Central Processing Unit) is the part that actually runs the programs you provide. Before a CPU can run a program it must be translated into a language that the CPU can understand. So all CPUs execute programs, but the program has to be in a language they can understand. An ARM CPU cannot understand instructions written for an Intel or AMD CPU and that effectively means that most off the shelf software (such as Microsoft Windows and games) cannot ru on a Raspberry Pi.

Okay so you're not going to be running Windows XP on the Pi, but does the ARM offer any advantages? The first advantage is that ARM CPUs draw much less power. This isn't overly surprisin as ARM really came from the embedded hardware industry where power usage and heat generation a a really big deal. In fact you'll find some form of ARM in almost every modern cellphone including the iPhone 6 and Samsung Galaxy S6. In fact they are one of the most widely used processors in the world and can be found in all manner of devices such as DVD players, appliances and even cars.

ARM CPUs also generate very little in the way of heat. If you look at the Pi itself, you will notice the CPU doesn't have a heat sink. If you look at any Intel or AMD CPU you can't fail to miss the hug cooling fan that it requires in order to prevent it burning out. Some people even user water cooling systems to keep their PC processors running at a reasonable temperature.

The last benefit is really a cost to performance ratio. For the vast majority of things, the real bottleneck is not CPU power but how fast data can be fed to it. CPUs have long been much faster than hard disks and even the bus that links all the computers components together can't keep up with even a modest CPU. So what do you get out of this? A low cost processor that almost certainly will do everything you need without the cost penalty.

WINDOWS 10 IOT

In the first edition of this book, we highlighted that although Windows 8 was shipping for ARM devices (most notably for tablets and phones) it would not run on the Pi. It really didn't seem all that likely that things would change in that area either as the Pi is basically a one operating system platform.

However two years on, it seems the tables have turned. Not only will you be able to run Windows 10 on a Raspberry Pi, but it will be officially support by Microsoft and, most importantly it will be free!

So what's the catch? Well, it's still in preview as I write this, and it will be the Internet of Things edition. That is, it will be very cut down and have limited features – you won't be able to run the full version of Windows 10 on your Pi. Still, you will be able to run Windows and gain access to main of the features that people use for building kiosks and data gathering devices and so forth.

As an added bonus, we will be putting together some chapters on Windows 10, just as soon as it comes out of preview. They'll be made available to you as online chapters!

1GB of RAM

The original version of the Pi had only a very modest amount of memory, just 256MB of RAM. Then the second revision of the model B was released and it double to 512MB. Today, the Raspberry Pi 2 clocks in at 1GB of RAM, which is really impressive for such a low cost device. Still, with laptops shipping with 8GB of RAM as standard these days, what can you really do with just 1GB?

Well the answer to that is a lot more than you might think. Remember, Windows 95 was able to operate with a couple of running programs in just 8MB of RAM (which conveniently enough is 128

times less than the Pi) and of course the good old Commodore 64 came with only 64KB (the Pi has something like 16,000 times that) and it was able to run thousands of games to entertain the masses (though admittedly not all at the same time).

So why do we have so much memory in new machines? Memory is cheap these days and althoug we could do lots of great stuff in very little memory, it's a skill that has gone out of fashion. Why spend so much effort on optimizing memory usage when it will most likely never matter? If 8GB of RAM costs $50, is it worth hours if not days of a programmer's time to save a few megabytes here a there? Probably not.

Remember, the Pi is meant as an experimentation platform, not as a general PC replacement and Linux (especially without a GUI) will run with plenty of memory to spare for all your programs. By keeping the memory to a reasonable minimum, the Pi is able to hit its price point without greatly hindering what you can do with it.

GPU (Graphics Processing Unit)

The GPU is really a specialized form of a CPU. A CPU is generic in what it can do and it tries to be good at everything it does, the classic jack-of-all-trades. A GPU on the other hand does one thing and one thing well. It is specifically designed to handle the intense mathematical calculations needed to render complex displays. This started off predominantly in 3D graphics rendering but has more recently gained traction in day-to-day computing where rather than computing graphics for display, a GPU can be harnessed to execute similar types of instruction for the user. For example, when an application such as Photoshop enhances a photograph, it applies an algorithm to the image and historically the CPU would do this. Today, Photoshop can offload that processing work on to the GPU which due to is very specialized design can do the work much faster – all without needing any assistance from the CPU itself.

The main reason why a GPU is important in a device like the Pi is that even with a modest processor, it can still handle high quality displays and decode high quality video streams. This makes the Pi useful as a media device as well as allowing for a full graphical display that still feels snappy even with a slower processor.

For the most part, the GPU is not something you will directly care about but by knowing that the heavy duty graphics work can be off loaded somewhat from the main CPU, you can be more confiden that the ARM that powers the Pi will be able to deliver enough brute force for your needs.

Ethernet Port (Model B only)

You might think that there's not really much to say about an Ethernet port. After all it's pretty straigh forward. You plug it in and you can access the network. If you don't have one, then you can't. Simple enough surely?

For me, the ability to connect to a wired network is essential. WIFI can often have issues and sometimes WIFI isn't even available (especially if you decide to turn your Pi into a WIFI access point). Although this feature costs an additional $10, in my experience not having the network card when you really need it will cost you much more than $10 in time, effort and general hassle. It is true you can add a USB network card to the Model A (and technically the card attached to the Model B is actually connected via USB) but then chances are that would cost at least $10 and then you'd have a USB device flapping in the wind. You might say to yourself "If it's USB then I can use it with my other devices too!" but in reality, you probably never will and you're far more likely to lose the damned thing altogether (right when you most need it) than it is to come to your rescue in a time of need.

However, one reason why you might decide on a Model A is that because it doesn't have these

extra components, it consumes significantly less power. This probably won't be a major concern for most users, but if you're planning on using the Pi in a battery powered product, then you would be very keen to lower the power requirements as much as possible.

Overall, my recommendation is to get the Model B, just because you get the network card. If you're absolutely 100% positive that you'll never ever need Ethernet (or you want the smallest possible power footprint), then there's probably little point paying the extra cash for this version.

USB

This isn't so much a feature these days as a true requirement. Almost all peripherals connect this way and the Pi is no exception. It will work with all standard USB keyboards and mice and assuming Linu has a driver for it, other USB devices as well (such as the Ethernet card in the Model B).

All models come with USB 2.0 support, although the Model A only has a single port, Model B ha two and the Model B+ as Raspberry Pi 2 have 4. As before, unless you have a specific reason not to, you should be looking to get yourself a Raspberry Pi 2.

GPIO Ports

GPIO (General Purpose Input/Output) ports are a very interesting and key addition to the Pi. Most people probably will never use these pins and most people will live a long and happy life having neve heard of them. They provide an easy way to connect hardware to your Pi that you can then control through software. If for example you wanted to add a thermometer or light sensor, you could build a device that connects to your PI via the GPIO ports.

If hardware projects don't really interest you then you can probably forget about the GPIO ports. However if you're looking to integrate your Pi with various bits of hardware or make your Pi the brai of some wacky invention, then GPIO ports will give you an easy way to do that.

Baked to perfection

By now you'll have picked up a pretty good appreciation for what the Pi is and what makes it special It was designed to be low cost so that everyone who wanted one could get their hands on one, but the designers have gone to great lengths to make sure that even though the Pi can't deliver everything a desktop can, it delivers more than enough to make it a fully functional computer laboratory not to mention a very nice server platform.

Whistle-stop Tour

So, what do we have in store for you in the rest of the book? The book is broken down into three core parts. In the first part I'll show you how to get up and running with the GUI. If you've never heard of GUI then this is definitely where you want to start. The Pi comes with everything you need but out of the box it needs a bit of tweaking. Don't worry, we'll get you up and running quickly and you'll soon be off exploring.

Part two takes you back to the command line and teaches you how to move around. Many people (myself included) spend the vast majority of their time at the command line. It's fast, powerful and always available (even when a GUI is not such as when you want to make a changes via the network) but it is a little bit different from the GUI that most people are used to. Fear not, we all have to start somewhere and in this section we'll make sure you're comfortable and at home on the command line

In Part three we actually start to do more interesting things. By this stage you're happy on the command line (which puts you ahead of most of the crowd) and now you want to delve a bit deeper and actually make the Pi work for you. We spend this section covering some of the great things you

can do and while we're at it we'll give you the solid foundation you'll need to do all of the hardware and software projects in Brendan Horan's "Practical Pi Projects".

Your first bite of Raspberry Pi

In Chapter one we look at what Linux is and why it's on your shiny new Raspberry Pi. We take a brie look at different "distros" and explain why not all "Linuxes" are the same. We take a closer look at th Raspberry Pi and what makes it just that little bit different and then we get your graphical interface u and running.

Surveying the landscape

Following on from where we left off in chapter one is a tour of the desktop and some of the fun thing that you can find. We take a look at the surprisingly wide range of software that comes preinstalled o the Pi and show you some of the things that it can do.

Getting comfortable

Now for something just a little bit different. You can use your Pi just like your other computer but now we're going to take it up a notch. In order to get the full benefit from Linux and to get your hand dirty with some very interesting projects (automatic cat detection and prosecution anyone?) you'll need to lift off the crust and start getting closer to the metal.

The first place to start is the command line where you'll be interacting with Linux on a very precise and powerful level. We'll be starting off slow and easing you in to what is likely to be a very alien environment. If nothing else you'll probably find it oh so quaint. We'll discuss the shell, where it came from and why it's important. We'll also look at a bit of the history involved as knowing wher it came from (while fascinating in and of itself) will help you get the full benefit from the experience

The file paths to success

Once you're up to speed on what the command line is, it's time for the most important subject. Most books start by telling you about all the shiny commands you can run and that's all well and good, but first you need some context.

I'll start you off by looking at the Linux file system and explaining the "everything is a file" philosophy that sits at the core of every Unix based operating system. This sounds a bit scary and we won't go into too much depth, but with this under your belt you'll be able to easily pick up new commands and make full use of them.

Essential commands

Now you know about the command line and how Linux lays out all your files, we'll actually showing you how to make Linux do things. I'll show you how to become root (does anyone remember the "power up" scene from He-Man? No?) and how to install new applications and tools. We will also briefly look at some useful command line tricks that will make you a keyboard demon. You will soon find that you can do things far more quickly on the command line than by clicking your mouse…

Editing files on the command line

Next on our tour de force is editing files. I will show you my two editors of choice. One (nano) is simple, easy and great for general use. I use it a lot for making small changes to config files. For mor serious heavy lifting, I prefer vim. It has all sorts of magic codes and key presses (really it's like playing dungeons and dragons – not that I ever have) and if you can hold all of the key-combos and commands in your head, well, you will never want to leave! For the rest of us, well, there's always nano…

Managing your Pi: from beginner to admin

Now we're starting to take a look at system services and what makes Linux tick. You'll learn that as smart as Linux is, it's not all that different from all the tools and things you've been running so far. You'll be able to apply your newfound knowledge and be able to start and stop services to your heart content.

Network services are the bread and butter of any server. Most people have used them from a user perspective but now it's time to experience them from the pointy end. We'll look at the usual suspect such as OpenSSH and the Apache webserver. After all that we'll have a little sit down and explain th very real dangers of running your own servers on the Internet. There are people out there who would quite like to gain entry to your nice little server and they do not much care who you are. Don't worry we'll provide a simple list of do's and don'ts that will put you ahead of the game.

A LAMP of your own: a Pi Webserver

We start off our project section with a classic, getting a full LAMP (that is Linux, Apache, MySQL and PHP) stack up and running on your Pi. We will show you why your Pi is an ideal little web serve and how it not only provides the perfect environment for development dynamic websites but also makes a great little portable web site demonstration tool!

WiPi: Wireless Computing

What's more fun than a tiny little network server? A tiny little network server that has no wires! We' show you how to hook up a USB wireless network adapter so your Pi can talk to the world without an wires. If that wasn't enough, we take it one step further and look at ways of cutting the cords altogether and getting your Pi to run off a battery.

The Raspberry sPi: Security cam and messaging service

Want to know who is sneaking into your room and stealing all of your pens? Your sPi has you (or you office) total covered! In this chapter we will show you how to hook up a webcam to your Pi and have it take snapshots of any movement in the room. Not only that but it will promptly email you a copy o the intruder in real time! Even James Bond doesn't have one of these…

Pi Media Centre

Last but by no means least we show you how to turn your Pi into your personal media center. We sho you how to stream video to your TV and how to act as an airport device for sharing music around the house. With high resolution streaming video, we really start to push the Pi and show what it is truly capable of!

Summary

We've covered a huge amount in this chapter already. We've looked at how computing has changed over the years and how this has not only had an impact on how we see and use computers in our daily lives but also how this has affected the way computers are perceived in education and the level and depth of knowledge that students are taught. We briefly touched on the Raspberry Pi Foundation and how they hope that the Pi will start to reverse this trend.

Next we looked at some of the more compelling reasons why you might want to get a Pi and why the Pi makes an ideal platform for development and experimentation.

We then looked at what the Raspberry Pi is at the hardware level and discussed the differences between your main computer and the Pi. We examined the trade offs and looked at why some of the decisions were made and what impact they will have on you as a user.

Last but my no means least, we rounded off the introduction with a whirlwind tour of all the thing
that we're going to cover in the rest of this book. Right then, let's head to the kitchen and make some
Pi!

Acknowledgments

Without a doubt, the true heroes of this book are the Raspberry Pi Foundation and the Raspberry Pi community because without either of them, we wouldn't be here. The Raspberry Pi Foundation, the group who bought us the Pi, is a nonprofit charity with the stated goal with the Pi was to reinvigorate the development spirit of the '80s and to provide an affordable platform for generations of youngster to get into the wonderful world of computing in a truly meaningful way. It has done so much to bridg the digital divide and bring computing to the common man, and for this it deserve our thanks.

If the Raspberry Pi Foundation is the heart, the Raspberry Pi community has to be the soul. These people have sought to provide the tools and support people need to get started with the Pi regardless skill level. They have invested countless hours (and dollars) in writing, fixing, porting, hacking, and testing things for the Pi. Please feel free to pay a visit to them at http://www.raspberrypi.org/ and s what we mean.

Contents