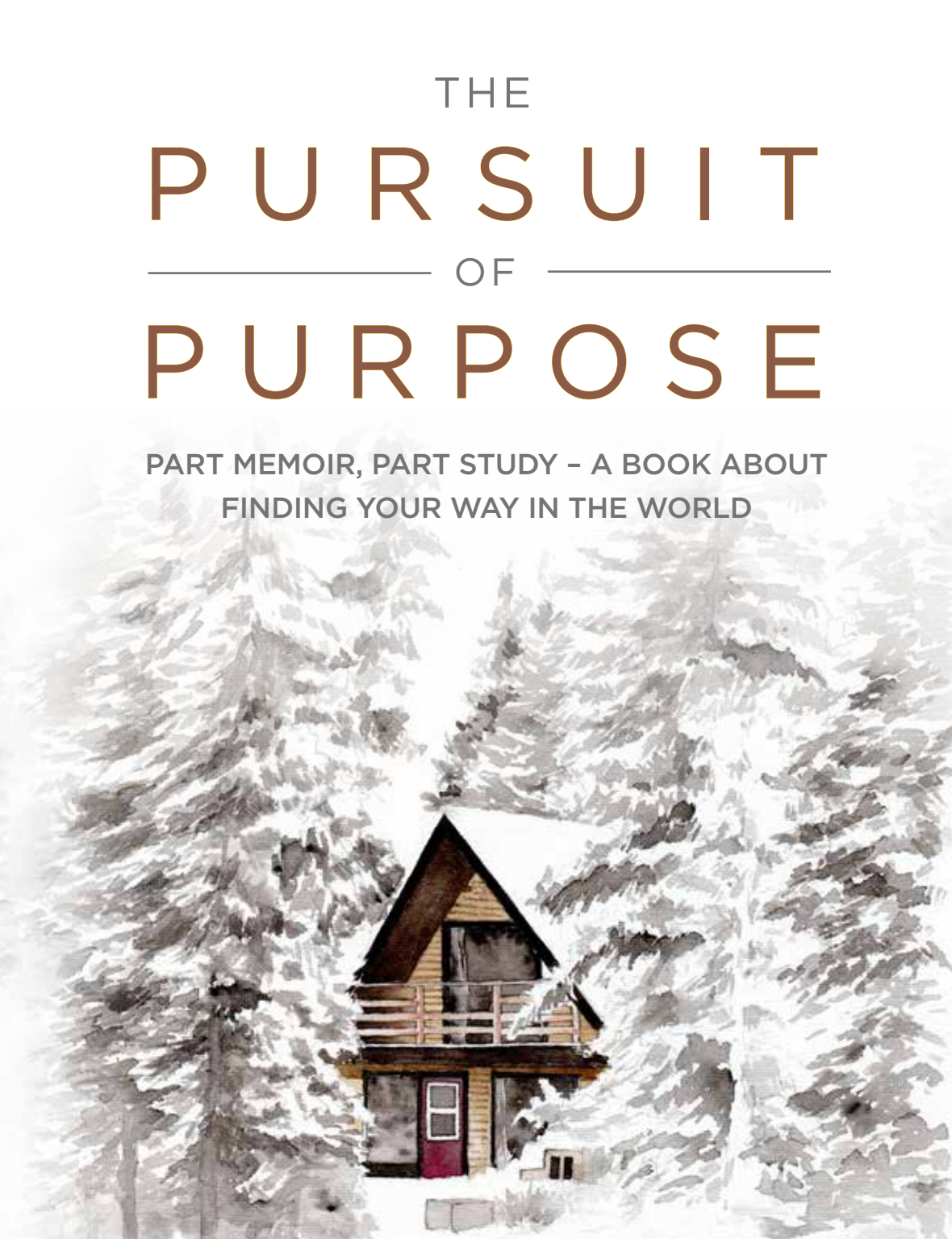
THE PURSUIT OF PURPOSE

PART MEMOIR, PART STUDY – A BOOK ABOUT FINDING YOUR WAY IN THE WORLD



KEN BANKS

Hello World!

Ideas often have a habit of popping up in the most unexpected of places or at the most inconvenient of times. If you're lucky you might be somewhere you can write them down, to make a record of them somehow, before they slip back into the depths from where they came. Most of the world's ideas go that way, vanishing before they take hold, often gifted to someone unable to capture the moment, or ignored by those with no intention of acting upon them in the first place.

Of course, being blessed an idea is just the beginning. Without execution they remain at best fleeting glimpses of what might have been, at worst mere scribbles on paper. I still have dozens of ideas I'm yet to do anything with, scattered around in everything from cupboard draws to computer folders. I'll likely go to my grave before I get back to most of them. Thanks to the Coronavirus pandemic and a succession of lockdowns, this book is no longer among them.

Throughout my life I've always had a good imagination, and I'm often invited to join meetings or working groups because of my perceived creativity, or ability to think outside of the box. Perhaps I do have a talent, but conjuring up ideas on demand isn't usually one of them. My brain just doesn't seem to be wired that way. Rather, my ideas tend to come to me unexpectedly, such as on long walks, when I'm in the shower, while I'm cooking, lying asleep in bed, or simply watching TV. Whenever I've found myself stuck writing this book, for example, I've often managed to conjure up a way forward when I've been furthest away from the keyboard. Fortunately, most of the time I managed to make a mental note of what came to me, be it the answer to a tricky opening or closing to a chapter, or ideas for a new one altogether.

So it was by pure luck, then, that the one big idea I was to have in my life came to me while I was sitting comfortably at home, an arm's reach away from pen and paper, and

not while I was standing in a gale on a remote hillside somewhere. Add the fact that African communication challenges were the furthest thing from my mind at the time, and my light bulb moment becomes even more of a mystery.

Indeed, that Saturday evening my thoughts were elsewhere as I lay on the sofa watching football on television, half asleep, with a near-empty bottle of Newcastle Brown Ale in my hand. Apologies in advance to anyone expecting a moment of inspiration on a par with Sir Isaac Newton, who had his a-ha moment as he pondered gravity while sitting under an apple tree. Or Archimedes who, taking a break from his study of mass, had his as he displaced water while getting into the bath. Newton and Archimedes were already investing huge amounts of time and brain power trying to understand those two particular problems, and it was very much on their minds when their big moments came. Their stories have since become a huge part of their legacy. Who doesn't think of apples when they think of Newton, or of Archimedes screaming 'Eureka!' as he jumped out of the bath before running naked down the street?

My moment couldn't have been more contrasting. Boring, even. I hadn't given a second thought to the communication challenges I'd seen as I travelled around Bushbuckridge a year earlier. Somewhere deep in the back of my mind I clearly recognised the problem, and possibly an opportunity, but I felt no motivation or obligation to do anything about it, and it had all become lost to the ravages of time. Why it suddenly came back to me that Saturday evening, at that moment, is something that confounds me to this day. But it did, and I'm eternally grateful for it. (It's a moment now immortalised in a short, animated National Geographic 'Live Curious' video, where a football bounces over a sofa with a bottle of beer precariously balanced on an armrest, as I explain the source of my 'big idea'.)

Within seconds I hastily put what was left of my beer down on the small coffee table next to me, and strode the short distance across the room to my desk, which sat tucked away in the far corner. Grabbing a pen and a scrap of paper I quickly jotted it all down. I didn't want to miss too much of the football, so I drew a quick diagram and labelled it up. A laptop computer, a cable and a mobile phone, and the odd scribbled word and an arrow here and there. That was basically it. I returned to the football, finished my beer, and within the hour was fast asleep in bed.

The following morning, after a lie-in and a quick, late breakfast, I sat down at my computer in my pajamas with a mug of freshly brewed coffee and decided to do a little research. I looked again at my rough notes from the night before. Was this new? Had it been tried before? If not, why not? Was it even possible? So many questions, and only one place to turn for answers – our old friend, the Internet.

It became immediately obvious there were plenty of platforms available that allowed you to send hundreds of text messages to people's phones from a computer, which is basically what I wanted to do. But every product I found needed three things that I knew would be a problem for organisations working in places like Bushbuckridge. Firstly, they relied on an Internet connection, something which was hugely problematic (it still is in many parts of the world, even today). Secondly, text message 'credits' needed to be purchased using a bank credit card, something else that none of the predominantly grassroots nonprofits or activists that I'd spoken to would have. Last but not least, because these products were all Internet-based, most were unable to receive replies from the phones they were messaging. You can't, after all, text the Internet – it doesn't have a mobile number. I wanted to create a genuinely two-way messaging solution, not something that was top down and one-way.

Remarkably, if my scrappily mapped out idea worked it would solve all of these problems, three problems which I believed, at the time, were the main stumbling blocks preventing text messaging from becoming a genuine tool for social change in developing regions of the world. After a couple of hours online I came to the conclusion that what I wanted to build didn't exist, not as I imagined it, anyway. Now I just needed to find out why. Perhaps it wasn't technically possible, in which case that would have been the end of it.

At the heart of my idea was a simple computer program that would run directly on a low-end, cheap laptop computer, one which would allow the adding of contact names and mobile numbers, and then the creation of groups to which those contacts could be added. The groups could then be selected and blasted with a single text message. The sender might want to inform community leaders in a particular village about the date and time of a proposed meeting, or share health information with a group of community health workers, or give market prices to groups of farmers to help them determine the best prices to charge for their crops. Once the message

was typed into the computer, the user would select the group to send the message to and the system would send commands down an attached cable, to a regular mobile phone at the end, which would then send the messages out one-by-one over the local mobile phone network. I knew I could probably write the software to handle contacts and groups, but what I didn't know was whether you could use a cable to transmit a series of commands to a phone to tell it to send a text message, and if so what those commands were. I never shied away from a challenge, and this was the sort of challenge I loved.

It was time for a break, something to eat, a little fresh air and then one final bit of research. Everything would hinge on what I found out after lunch.

Years earlier, during my Clifton Computer days, I'd learnt how to use a piece of software called Carbon Copy. This nifty little program enabled me to connect with other people's computer systems remotely, over a dial-up modem and phone line, allowing me to carry out most technical support without having to leave my desk. Thanks to Carbon Copy I became familiar with the process of setting up and configuring modems, and learnt something known as the Hayes command set, a simple language that tells modems what to do and how to do it. These skills, picked up more than ten years earlier, would prove invaluable over the next few days as I set out to prove my idea.

Over the course of that morning things had fallen neatly into place, but I still didn't know whether a regular mobile phone, when connected to a computer with a cable, would act like the kind of modem I was used to, or whether it would act like a modem at all. And, if it did, whether any Hayes commands existed to tell it how to send and read a text message. I randomly selected a mid-range phone from the Nokia website, downloaded the technical manual, and began to read. Towards the back, hidden in an appendix, there they were, references to Hayes commands that could tell the phone not only how to send a text, but also how to read one back from the phone's inbox. My heart picked up a few beats. I felt so very close to proving the viability of my idea. All that remained was to try it out for myself, with real equipment.

I searched for a second-hand Nokia phone and the right cable to connect it to my Dell computer. This was early 2005, so I wasn't spoilt for choice with online shopping sites.

In fact, only eBay had what I needed, and fortunately I was already a member. After cross-checking available phones with the Nokia website – I needed to be sure that I not only bought one that supported Hayes commands, but one that would be available to users in developing countries – I bought a second-hand Nokia 6100, a CA-42 cable and a pay-as-you-go SIM card to use for testing. I hit the 'Pay' button, and logged out. Satisfied with my day's work I closed my computer, pushed all my notes to one side and got on with what was left of my weekend.

Over the course of the next few days the equipment gradually arrived. It was mental torture, the greatest puzzle of all arriving piece-by-piece, but by the end of the week I had everything I needed laid out neatly on my desk. My moment of truth was here. Everything would now hinge on what happened during the next few minutes.

I slid the SIM card into the Nokia phone, replaced the battery and back cover and powered it up. It had a couple of bars of power remaining, just enough to do the tests I needed. I connected the CA-42 cable to the base, waiting for the reassuring click, and screwed the other end into the 9-pin serial port at the back of my computer (these were the days before everything came with more familiar USB connections). I logged into Windows and opened Terminal, a small Microsoft Windows program that allows users to communicate with any devices physically connected to the machine they're using. The cable was plugged into the only communications port I had on my Dell, so I knew the phone sat at the end of COM1. Using a combination of my own memory, commands I found online and notes from the Nokia technical guide, I changed a couple of default settings and, after a minute or two, was ready to go.

The first thing I needed to do was to see whether the phone could be 'seen' by Terminal. If not, nothing would happen no matter what commands I sent it. The quickest way to do this was to send the briefest of commands – the letters 'AT' in fact – down the wire and wait to see if the device responded. In simplest terms, it's the Hayes equivalent of 'is anybody there?' If something is there, and it understands that it's been asked to say hello, it will respond with a short but reassuring 'OK'.

I held my breath as I typed 'AT' into Terminal, before hitting the Enter key on the keyboard. Immediately, without a second's hesitation, the phone responded. There it was, two bold letters that said it all. 'OK' appeared on the computer screen. My breath

quickened a little more. The phone was there, and the computer could not only see it, but could talk to it as well. Just one step remained. Could I instruct the phone to send a text message using commands I'd type into Terminal, and then read one back from the phone onto the computer screen? This was the biggest and most important test of all, and the further I got the more disappointed I knew I'd be if I fell at the final hurdle.

Using the Nokia reference manual, I looked up the command to tell the phone what number I wanted to send the text message to, which happened to be my Sharp GX20, and sent it down to the Nokia phone. Next, I followed it up with another command containing the actual text I wanted to send in the message, 'Hello World!' (it's coding tradition to use 'Hello World!' as an opening message the first time you try something like this). Then it was time for the final command, the one to tell the phone to send the message. By now the anticipation was killing me. Breath well and truly held, I hit the Enter key one last time.

The blinking cursor on the computer came back with 'OK', followed by a pause, a pause which felt like an eternity. I stared nervously at the GX20 screen, waiting for any kind of response. It was all or nothing. Had I done everything right? Was the message going to arrive? Was I just wasting my time?

It turned out I wasn't. The message arrived, and I couldn't contain my excitement. The reassuring series of beeps from the Sharp told me it had worked, and I read the message back. 'Hello World!' indeed. I hit the reply button on the phone, typed a short message and hit the send button. A few seconds later the Nokia, still sitting on my desk with its cable attached, beeped reassuringly. I could now see my reply on its little monochrome screen. This was fantastic! But now the big question. Could I get my Dell computer to read it? I dived back into the technical guide and, a couple of Terminal commands later, and a little more holding of breath, the message appeared in plain text on the computer screen.

I couldn't believe it. I was over the moon. Ecstatic. It felt like some sort of magic trick to have been able to send and receive a text message without having to physically touch a phone, and to do it all via a series of commands on a computer keyboard. Less than a week after dreaming up the initial idea, I'd done everything I needed to prove it was, indeed, possible.

I sat down with a cup of coffee and, despite the caffeine rush, managed to come slowly back down to earth. I couldn't help but imagine all the things I could get a system like this to do, and all the people in all the places it might help. All that was left now was to write the software to run on the computer, software which would drive everything for the user. For this to work they'd need to be shielded from all the technical stuff happening in the background, and all the commands and configuring I'd just done manually. This was an exciting proposition, although I'd not done any coding of any real significance since leaving Jersey for university. I'd never written a Windows application before, either. I didn't know it yet, but there was going to be a steep learning curve ahead of me.

Before I went much further I decided to share what I was tentatively calling 'Project SMS' with Karen and Simon, who I was now working with on a few smaller, independent projects. I typed out a one-page concept note explaining what I hoped to build, what it could do and where it might be useful, and highlighted the differences between what I was planning and what already existed. I included a hastily cobbled-together image of a laptop with a phone and a cable attached to help them picture what the whole thing might look like. Karen was still in touch with her contacts at Vodafone, and we approached one in a personal capacity for help. If I was going to develop this piece of software then I would need a laptop of my own to write it on, a copy of Microsoft Visual BASIC to write it with, a few technical guides and a couple of different test phones and GSM modems on which to test it. We added a bit extra to the budget to cover some of my time, and settled on £10,000 which we figured would be enough to get Project SMS off the ground.

Things moved quickly, and in more ways than one. Simon did what Simon does best and, knowing I wasn't happy with my makeshift project name suggested we call it 'FrontlineSMS' (Simon was the one who came up with the name wildlive!, too). The new name was a perfect fit, and immediately stuck. The FrontlineSMS domain was bought a little later in March, ready for a website that we'd need to build when we eventually launched. The small amount of seed funding we were hoping for came through, and soon I was out shopping for a laptop, researching phones and modems and taking my first tentative steps into the world of Windows programming. It would be another few months before coding would begin in earnest, though. I did, after all, have other paid work on at the time and didn't want to launch myself into FrontlineSMS until I'd done

all the necessary research, laid all the groundwork, and was able to focus my attention fully on the task ahead. My apprehension was understandable. I wasn't entirely sure how things were going to work out, if they were going to work out at all. Would I even be able to write the thing?

FrontlineSMS development was pushed to the back of my mind over the next few weeks and months, but as I went about my daily business I continued to seek out opportunities for learning and for partnerships, or to join random dots I thought might help me, come summer, to hit the ground running. I became something of an expert at opportunistically dropping FrontlineSMS into random conversations and, one morning, my approach paid off in spectacular fashion.

That March I was invited to an event at Microsoft's main campus in Cambridge where I did what I was increasingly asked to do, and spoke about the potential social benefits of mobile technology. During the break I bumped into one of the product managers responsible for Visual BASIC. It was a moment of pure serendipity. Andrew was friendly and approachable, and the fact he liked my earlier work, and my talk, certainly helped motivate him to help with what I was planning next. At that time I hadn't even started exploring Visual BASIC, the programming environment I'd decided to use for FrontlineSMS, but I knew it was likely I'd need expert help, most likely at crucial moments. Cheekily, I asked for his help, and remarkably Andrew offered to make himself available and share any questions I had with the Visual BASIC developer team. We exchanged email addresses and, true to his word, he and his team were brilliant when coding finally began that summer.

Days and weeks passed by and the summer months were soon upon us. Elina, who was by this time studying full time at university in Sussex, had been offered a job with a family friend back home in Finland throughout August. Over a drink one warm, sunny evening in the beer garden at the Prince of Wales, our village pub, we talked through our options. Elina leaned forward towards the shady half of the table.

'So, Jussi has offered me some work over the summer, and I'm wondering if we could both go and turn some of it into a holiday.' She hadn't been home for a couple of years and was keen to go back to catch up with family and friends.

'Well, I need the time, and a load of headspace, to tackle this new FrontlineSMS project, so that could work', I replied. I knew Forssa well enough, and

knew it would be quiet and relaxing, allowing me to focus on my coding. 'How long are you thinking?' I asked.

'Maybe five or six weeks? University is closed so I have as much time as you can get, to be honest.'

'Okay, let me check tomorrow. If August looks clear, we should just go ahead and book the tickets', I answered with a smile, empty glass in hand. 'Another drink?'

It was a no-brainer, to be honest. Elina needed the money, and I needed time and space to lock myself away and tackle FrontlineSMS, so heading to her family home in Forssa for a few weeks felt like a win-win. I would have the house to myself as everyone would be out working all day. We went ahead as planned, and a few weeks later we arrived in Finland, back to the town I'd left in frustration exactly four years earlier.

Laptop, manuals, cables, phones, modems (and, more importantly, coffee) to hand, I set up shop on an old mahogany kitchen table which sat in the back corner of their lounge next to the window, giving me a wonderful view of the fields and forests outside. The room was typical of most Finnish houses, with an open-plan living space leading into the kitchen, all decked out with wooden flooring. Pushed up against the walls was the most basic of furniture, with plain wallpaper acting as a backdrop to scattered photos of family members – past and present – which were dotted around in the sort of old fashioned picture frames you usually only find in antique shops or museums. On the wall above the sofa hung a large picture of a lion resting under a tree, painted by Chiti, a welcome and constant reminder of my time in Zambia. A little further along sat an old, neglected, out-of-tune piano which Elina used to play occasionally as a child. The lounge, basic as it was, turned out to be an ideal place to work and, shortly after we arrived, I settled down into what would become a five-week coding marathon. It's hard to believe that when I sat at that old table for the first time I had no idea what I was taking on. By the end I'd have exceeded all my expectations, and would get on the plane home with not only a solid version of FrontlineSMS to share with the world, but also a fully functioning website.

The coding was, as expected, a challenge, but the kind of challenge I enjoyed. I was now living in the world of Microsoft Windows, and I had to think about things in an entirely different way to what I was used to. In BASIC programming of old, you always knew which line of code the computer was running, and you'd simply give the

system instructions about what to do next, a decision usually based on what the user entered on the keyboard. If, for example, you were just waiting for something to be pressed, the system would simply sit at that line of code and wait. Visualising your code running was a relatively straightforward and simple thing to do.

Windows turned out to be a different beast altogether. When a user sits in front of a typical Windows application, there are usually a number of things they can choose to do at any one time. They might click on Cancel to go back, or click inside a text box to type something, or scroll down a list and select an entry, or click on a drop-down menu, or drag something somewhere. Basically, after laying out the screen your program needs to just sit and wait and, depending on what the user decides to do, different bits of code in many different places might be called to run. Because of this, Windows is known as 'event driven', and it was quite different from anything I had used or coded in before.

In all, I had to design about a dozen screens for FrontlineSMS, thinking carefully about each one and then deciding where on the screen each of the boxes, buttons and lists should go. This was the most enjoyable part. Dropping different elements onto a page and sizing and moving them around until they all looked right wasn't that difficult, and I always fancied myself as a bit of a designer. But that's where the fun ended. I also needed to think about how I'd 'string' all these different screens together, how I would pass information between them, and the specific function each had to carry out. And I had to do it in a way that would be immediately obvious to the users, many of whom would have little experience with computers, or messaging software come to that.

Remarkably, I had no detailed notes, no plans, no diagrams, no list of functions. I broke all the rules and just built it as I went, relying on intuition and a combination of gut feeling and luck. Three years later I would be interviewed by a Nokia engineer, and he would be disappointed to learn that I hadn't followed any established 'design principles' in my work. I didn't even know there were any. It was seen as something of a miracle that I'd built something as effective as FrontlineSMS without, on the surface at least, really knowing what I was doing.

Every time I got stuck and couldn't find the solution in one of my hefty Visual BASIC manuals, I'd turn to the Stack Overflow website, an online community of developers

who seem, at some point, to have answered every conceivable question about every conceivable computing problem in history. If, by some miracle, there wasn't anything useful there I'd down tools and go for a leisurely stroll through the endless forest that lay on my doorstep. Aided by the warm sunshine and gentle breeze, these walks did wonders to clear my mind. They were spiritually refreshing, too, a great way to escape and think about anything other than text messaging and coding. Putting distance between me and my keyboard usually did the trick, and I'd often return to my laptop and dive straight back in with a new lease of life.

When it didn't work I'd reach out to Andrew at Microsoft, my knight in shining armour. Usually within a day I'd have a reply, forwarded on from one of the team who provided the sort of high-level technical support that, without Andrew, I could have only dreamed of. I would reconnect with him many years later, during one of the pandemic lockdowns, and let him know how incredibly valuable his contribution to the FrontlineSMS story was. It may have been nothing to him, but it meant the world to me.

The more I progressed and battled my way through each major coding obstacle, the more my confidence grew. Just a couple of weeks in and things were already beginning to take shape, and I was really enjoying writing software again. I even managed to relax a little, allowing myself the occasional daytime distraction.

Summer that year coincided with one of the biggest sporting rivalries in the cricketing calendar – the Ashes series between England and Australia – and radio commentary was streamed on the BBC website. Both sides would duel over a small brown urn which, according to legend, contains the ashes of a wooden bail, burnt after a match in 1882 when Australia beat England on English soil for the first time. The Ashes represented the 'cremated body of English cricket' and the sides have fought over the tiny four-inch urn ever since. That summer the series was in England, and I listened online to the daily commentary as I merrily coded away. It was great company and, at times, a great distraction, particularly when Australian wickets fell. Occasionally I sent messages to the BBC Cricket commentary team, and twice they were published on the BBC website. The messages are still there, a nice reminder of a summer I will always look back on with great fondness and pride.

One morning in late August I settled down as usual with my coffee when an email popped up in my inbox. It was from a journalist at the Charity Times who was writing an article about how non-profits were using mobile technology. She asked if she could send over a few questions about wildlive! and my work in South Africa. I agreed and, opportunistic as ever, in my reply I managed to squeeze in a mention for FrontlineSMS, even though it wasn't finished yet.

A couple of days later my phone rang.

'Oh, hi Ken. We exchanged emails earlier in the week about this Charity Times article, and I wanted to quickly check something with you, if that's okay?'

'Sure. Happy to help. Fire away.'

'So, I'm drafting up a paragraph on FrontlineSMS. Can you remind me when you hope to have it ready?'

'I'm aiming for October,' I replied, confidently.

'Okay, great. And one last thing, where can people go to find out more?'

It was then that it dawned on me. Website. I had no website! Where would people go to get more information, or to sign up to try it out? I'd just been presented my first big marketing opportunity, and I was about to blow it.

'Oh, that would be www.frontlinesms.com.'

'Great. Thanks, Ken. I'll let you know when the article goes live, hopefully in the next week.'

I hung up. Plans for further coding were immediately put on hold as I frantically put together the first FrontlineSMS website. I had all the content, albeit in my head, so I just needed to add some structure to it, add a few images here and there and get it live. One important thing was missing, though – a banner image for the top of the home page. Short of ideas, I quickly took a photo of the field and forest outside my window, and by sheer luck it was a perfect fit. It still brings a smile to my face whenever I see archived versions of that very first website, a reminder of my view each day as I coded away. By today's standards the website is nothing to write home about, but I don't think it was bad at all for 2005. At least now I had one and, as it turned out, it did the job perfectly.

The combination of coding and bouts of walking, relaxing and drinking Finnish beer on the banks of some of the many local freshwater lakes, all in the warm, endless daylight,

turned my trip into the perfect working holiday, and progress with FrontlineSMS was swift. Five weeks always felt incredibly ambitious to write a messaging platform from scratch, especially with no plan, but as September arrived it was as good as complete, and I'd managed to throw in a website for good measure, too.

Crucially, because I'd written everything myself, it would be simple to fix anything that didn't work, or add anything I may have missed, without the need to hire anyone to do it. Being able to support the ongoing technical development of the platform, as well as provide technical support to many of the early users, turned out to be crucial to its early survival and later success. It would be two years, during which time I ran it all as a hobby, before any significant funding would come my way. The importance of keeping things lean, particularly at the beginning, turned out to be one of the key bits of advice I would share with other would-be founders and innovators for years to come. You'll find it mentioned later in this book, too.

We returned home to our one-bedroom flat during the first week of September and took a couple of days off before Elina headed back to university to prepare for her second year of study. I would then be free to get back to my consultancy work, adding any finishing touches to FrontlineSMS during the odd evening and weekend. As it crept ever closer to completion, I began thinking about how I'd get word out. I tentatively contacted various groups and journalist friends who I thought might either be interested in trying it, or be able to help share news of the launch. FFI would have the honour of hosting the first ever public demonstration when I planned a lunch time talk for 29th September, an event that felt particularly poignant given it was my work there that had inspired the idea in the first place.

As I continued to think about exciting ways of promoting FrontlineSMS, I stumbled across a BBC news report about a brilliant new website launched by a 21-year old British student by the name of Alex Tew. It was something I quickly decided FrontlineSMS needed to be a part of. Alex was short of money to cover his university fees, and came up with the idea of building a web page exactly one million pixels in size, selling each pixel (or dot on the screen) for a dollar. On 26th August 2005, the Million Dollar Homepage was born and it quickly captured the Internet's imagination. By October it was already well on the way to filling up, so I waited no longer and took the plunge, purchasing 200 pixels for FrontlineSMS. It was the only money I would

spend promoting the platform in the early years and, although the Million Dollar Homepage might not have been the best strategic fit, I knew it would become part of Internet folklore. The last block of one thousand pixels was put up on eBay in January 2006, selling for $38,000 and signalling the end of Alex's incredible adventure. The website is still online today and, hidden among the hundreds of images, you'll find two small squares carrying the graphic of a tiny mobile phone, an image that provides one of the earliest links to what was then the first ever FrontlineSMS website.

By now I was only three weeks away from my own launch, and the coding for FrontlineSMS should have been done and dusted. But this was no ordinary project, and I guess I was no ordinary programmer. As a sign of the organic nature of it all, an idea came to me randomly one afternoon as I sat on the top deck of a London bus. As it stood, FrontlineSMS allowed you to add contacts and put them into groups, and to send and receive messages to and from those groups, and it did that brilliantly. What was lacking was some sort of automation, something interactive. And that's what came to me on the bus that day. What if I added an 'AutoReply' feature, I thought to myself. Users could text in a specific keyword, and then FrontlineSMS could automatically respond with a predefined message, anything from a bit of health advice, or the opening hours of a clinic, or the price of cassava. It sounded like something fun to try and write but, more to the point, it sounded pretty useful.

I immediately realised the value of adding this sort of functionality, so that evening I got to work on adding what, this time, really would be the last big change. I was done by the weekend, and it worked a treat. I could now leave FrontlineSMS sitting unattended on my desk, send in a text message from anywhere and have it respond without any human intervention. I didn't realise it at the time, but this one piece of automation was to prove incredibly powerful when increasing numbers of community groups began taking an interest in the platform over the coming years.

October 31st, 2005. The big day. With a mixture of excitement and anticipation, I pulled up a chair and sat down at my Dell computer, typing out an email to all my contacts, friends and supporters with the subject 'The FrontlineSMS system is now ready for trial'. Eyes fixed on the screen, I took a deep breath, allowed myself a wry, contented smile, and hit 'Send'. A random idea that had come to me in that very same room,

just ten months earlier, was now a reality. There was no going back, it was out in the world, and it was now up to the world to decide what would happen next, if anything.

Around the time of the launch I was in the early stages of a three-month contract with TTPCom, a Cambridge-based technology company that hired me to help test their mobile phone operating systems. I'd found the job randomly, casually glancing in the window of one of those recruitment agencies in Cambridge city centre a couple of weeks earlier. It was fun, and it paid well. The TTPCom engineers, who sat a few metres away, would do their usual software development and testing, and then hand everything over to me and two colleagues. Our job was to run through a series of tests to make sure everything worked as expected, and if it did we'd then move on to trying out random things to check how stable the code was. Because of my own programming background, I became particularly good at figuring out ways of crashing the phones, a success rate that did little for my popularity among my coding colleagues.

While they scratched their heads trying to figure out what I'd done and how I'd done it, there was little else for me to be getting on with. So I decided to take advantage of this unexpected free time and turned my attention to promoting FrontlineSMS. I'd search for anyone and everyone online who was involved with technology in Africa, or mobile phones, or activism, or humanitarian response. I'd email journalists, academics, nonprofits and technologists and share the launch of the platform, asking if they wanted to give it a go, or if they could help spread the word, offering myself up for interview in the process. There really wasn't anything like it at the time, and the origin of the idea in Southern Africa and the simple, offline grassroots focus made it a relatively easy sell at a time of growing interest in the use of mobile phones in global development.

The first person to break news of the launch was Mike Grenville. Mike ran one of the best and most popular SMS-focused websites at the time, 160 Characters. Having him share the news was a big deal, and Mike and I would later become friends. I was also asked to write a guest post for MobileActive, a relatively new and rapidly growing community of activists and practitioners interested in the potential of mobile phones for social change. These two articles, more than any, gave my outreach efforts a real boost, pushing the news into precisely the kinds of places I was hoping for.

One of those places was Zimbabwe. Reading the MobileActive piece from her office in central Harare was Brenda, co-founder of one of the country's most prominent civil society organisations, Kubatana. Against all the odds, Bren and her co-founder, Bev, had been actively and very publicly promoting human rights in the country, despite the obvious risks they took working right under the nose of the Mugabe regime. Kubatana had been something of a pioneer in the use of email to share news and information with fellow citizens, but they recognised the obvious limitations given Zimbabwe's weak technical infrastructure, poor Internet penetration, and low computer ownership. They had been taking more and more of an interest in mobile phones, and SMS in particular, as a way of spreading their message further.

Zimbabwe is often associated with the endless turmoil and violence of President Robert Mugabe's 30-year grip on power, but it wasn't always that way. Geographically, Zimbabwe is a beautiful, landlocked country, sharing its borders with South Africa, Botswana, Zambia and Mozambique. It was first colonised in the 1890s when white settlers arrived, and the country remained largely under colonial rule until 1964 when Ian Smith became Prime Minister. His first act was to demand independence from Britain and, when this was denied, he unilaterally declared it anyway, causing international outrage. At the time, Rhodesia (as it was known) was home to just 200,000 whites compared to four million blacks. Smith believed that black majority rule would never happen 'in a thousand years'. How wrong he'd turn out to be.

A certain Robert Mugabe had become increasingly active politically during the 1960s, a period of rising African nationalism, and as leader of the Zimbabwe African National Union (ZANU) he played a key role in the struggle for independence, a struggle that would ultimately lead to the deaths of over 30,000 people. Despite the loss of life the struggle eventually paid off and, in 1980, the country was finally granted independence from Britain. Robert Mugabe was declared Prime Minister. Seven years later he would change the constitution and become President, a position he would hold with an increasingly iron grip for the next 30 years.

At the time of independence, Zimbabwe (as it was now called) held great promise. Often referred to as the 'Jewel of Africa' it boasted an abundance of wildlife, and was one of the few African countries to be home to all of 'The Big Five'. And let's not forget Victoria Falls, one of the seven wonders of the world. The soil was rich and the

country was a net exporter of food. Earlier modernisation meant they had good roads and pavements, solid infrastructure, a number of airports and an education system that was the envy of the continent. As independence dawned on Friday 18th April, 1980, the future looked bright.

How things would change. Zimbabwe's promising start slowly eroded away, to be replaced by years of corruption and mismanagement. An economic crash followed, leading to a crisis of faith in Mugabe's leadership. Despite controlling much of the machinery of government he suffered a humiliating defeat when, in 1999, his proposed constitutional reforms were roundly beaten in a national referendum. To make matters worse a new party, the Movement for Democratic Change, or MDC, was emerging as a rare and very real threat to his hold on power. When black farmers started intruding on, and eventually seizing, white-owned farms as part of a growing dispute over land, Mugabe saw it as a way of regaining the initiative. Not only did he do nothing to stop it, he actively encouraged it. International donors pulled funding in protest, leading to a decline in foreign currency reserves, food shortages and further economic collapse. As if that weren't enough, Mugabe then restricted freedom of the press, and ramped up his blame on Britain and the international community for Zimbabwe's woes. Despite the chaos, Mugabe's party claimed victory in the March 2005 elections with two-thirds of the votes. The MDC cried foul. That summer, tens of thousands of shanty houses were raised to the ground as part of a government 'clean up operation' that, not unsurprisingly, targeted opposition MDC strongholds.

This was the scene when Kubatana reached out to me on 15th November asking about FrontlineSMS. Having followed Zimbabwe's gradual collapse – it was a staple on BBC News at the time – I was delighted to hear that my fledgling platform resonated with an organisation doing such amazing work in one of the most challenging of places. I immediately warmed to Bren and Bev, and worked with them over the next couple of weeks to get them up and running. In no time at all they were using FrontlineSMS not just to send news updates around the country, but to collect news from communities at the sharp end of the violence. I remember thinking to myself at the time that if Kubatana were the only people to use FrontlineSMS, I'd be happy.

As it turned out, there would be many, many more.