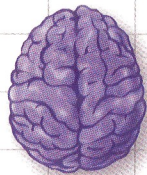
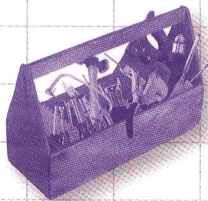


A Brain-Friendly Guide

Head First Programming

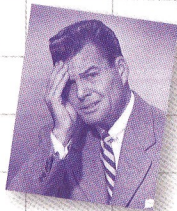


Load important coding concepts straight into your brain



Add methods, functions, and objects to your toolbox

Avoid embarrassing mishaps with input/output



A learner's guide to programming, using the Python language



Process your data like a pro

Build a functional and attractive graphical application



Learn how to automate repetitive tasks



O'REILLY®

Paul Barry & David Griffiths

Table of Contents (Summary)

	Intro	xxiii
1	Starting to Code: <i>Finding Your Way</i>	1
2	Textual Data: <i>Every String Has Its Place</i>	37
3	Functions: <i>Let's Get Organized</i>	77
4	Data Files and Arrays: <i>Sort It Out</i>	113
5	Hashes and Databases: <i>Putting Data in Its Place</i>	145
6	Modular Programming: <i>Keeping Things Straight</i>	177
7	Building a Graphical User Interface: <i>Going All Goopy</i>	215
8	GUIs and Data: <i>Data Entry Widgets</i>	257
8½	Exceptions and Message Boxes: <i>Get the Message?</i>	293
9	Graphical Interface Elements: <i>Selecting the Right Tool</i>	313
10	Custom Widgets and Classes: <i>With an Object in Mind</i>	349
i	Leftovers: <i>The Top Ten Things (We Didn't Cover)</i>	385

Table of Contents (the real thing)

Intro

Your brain on Programming. Here *you* are trying to *learn* something, while here your *brain* is doing you a favor by making sure the learning doesn't *stick*. Your brain's thinking, "Better leave room for more important things, like which wild animals to avoid and whether naked snowboarding is a bad idea." So how *do* you trick your brain into thinking that your life depends on knowing Programming?

Who is this book for?	xxiv
We know what you're thinking	xxv
Metacognition	xxvii
Bend your brain into submission	xxix
Rcad me	xxx
The technical review team	xxxii
Acknowledgments	xxxiii

starting to code

Finding your way

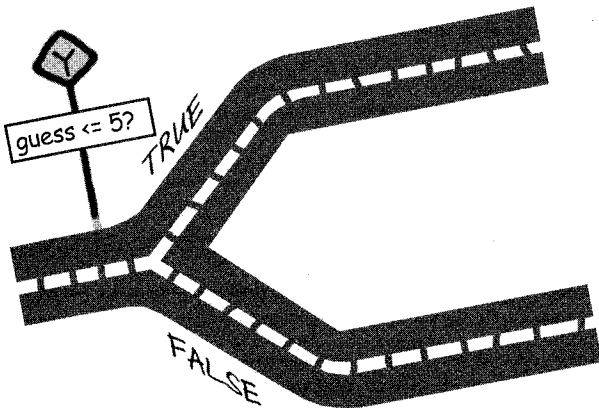
1

Writing programs gives you the power to control your PC.

Almost everyone knows how to *use* a computer, but few people take the next step and learn how to *control* it. If you use other people's software, you will always be limited by what other people think you want to do. Write your own programs and the only limit will be your own imagination. Programming will make you more creative, it will make you think more precisely, and it will teach you to analyze and solve problems logically.

Do you want to be programmed or be the programmer?

Programming lets you do more	2
So how do you run your code?	5
Create a new program file	6
Prepare and run your code	7
A program is more than a list of commands	12
Codeville: Your program is like a network of roads	13
Branches are code intersections	14
if/else branches	15
The Python code needs interconnecting paths	20
Python uses indents to connect paths	21
Loops let you run the same piece of code over and over again	28
Python's while loop	29
Your Programming Toolbox	35



textual data

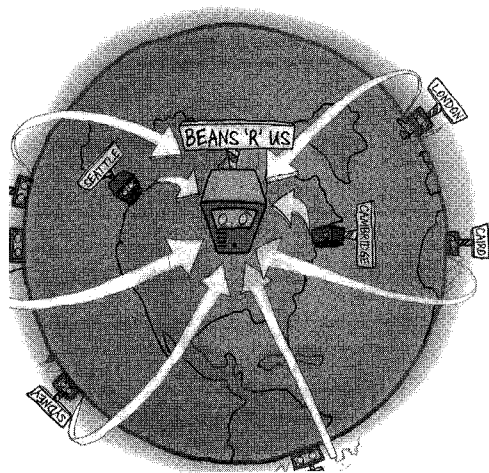
Every string has its place

2

Imagine trying to communicate without words.

All programs process data, and one of the most important types of data is **text**. In this chapter, you'll work through the basics of **textual data**. You'll automatically **search** text and get back **exactly what you're looking for**. Along the way, you'll pick up key programming concepts such as **methods** and how you can use them to **bend your data to your will**. And finally, you'll instantly **power up your programs** with the help of **library code**.

Your new gig at Starbuzz Coffee	38
Here's the current Starbuzz code	39
The cost is embedded in the HTML	41
A string is a series of characters	41
Find characters inside the text	42
But how do you get at more than one character?	43
The String Exposed	48
Beans'R'Us is rewarding loyal customers	50
Searching is complex	52
Python data is smart	54
Strings and numbers are different	64
The program has overloaded the Beans'R'Us Server	67
Time... if only you had more of it	68
You're already using library code	69
Order is restored	74
Your Programming Toolbox	75



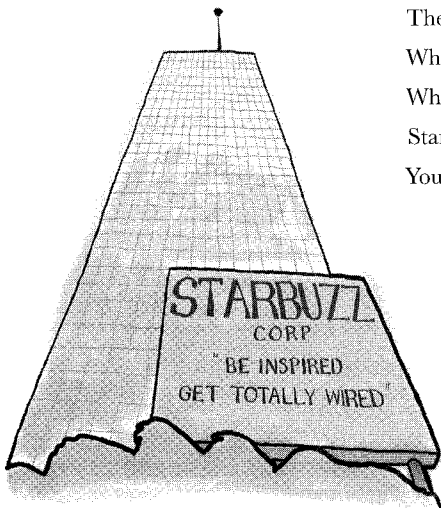
functions

3

Let's get organized

As programs grow, the code often becomes more complex. And complex code can be hard to read, and even harder to maintain. One way of managing this complexity is to create **functions**. Functions are **snippets of code** that you use as needed from within your program. They allow you to **separate out common actions**, and this means that they make your code **easier to read** and **easier to maintain**. In this chapter, you'll discover how a little function knowledge can **make your coding life a whole lot easier**.

Starbuzz is out of beans!	78
What does the new program need to do?	79
Don't duplicate your code...	81
Reuse code with functions	82
Always get things in the right order	84
Return data with the return command	87
Use the Web, Luke	93
The function always sends the same message	94
Use parameters to avoid duplicating functions	96
Someone decided to mess with your code	102
The rest of the program can't see the password variable	104
When you call a function, the computer creates a fresh list of variables	105
When you leave a function, its variables get thrown away	106
Starbuzz is fully stocked!	110
Your Programming Toolbox	111



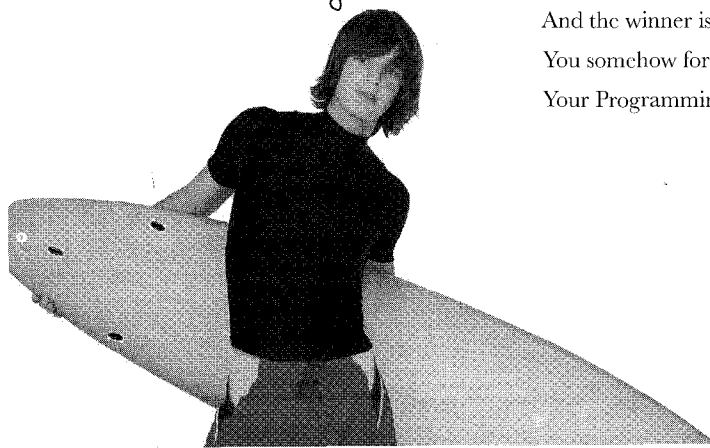
data files and arrays

Sort it out**4****As your programs develop, so do your data handling needs.**

And when you have lots of data to work with, using an individual variable for each piece of data gets really old, really quickly. So programmers employ some rather awesome containers (known as **data structures**) to help them work with lots of data. More times than not, all that data comes from a file stored on a hard disk. So, how can you work with data in your files? Turns out it's a breeze.

Surf's up in Codeville	114
Find the highest score in the results file	115
Iterate through the file with the open, for, close pattern	116
The file contains more than numbers...	120
Split each line as you read it	121
The split() method cuts the string	122
But you need more than one top score	126
Keeping track of 3 scores makes the code more complex	127
An ordered list makes code much simpler	128
Sorting is easier in memory	129
You can't use a separate variable for each line of data	130
An array lets you manage a whole train of data	131
Python gives you arrays with lists	132
Sort the array before displaying the results	136
Sort the scores from highest to lowest	139
And the winner is...?	142
You somehow forgot the surfer names	143
Your Programming Toolbox	144

Hey, dude, it's
Chapter 4... time
for a break - let's
catch some waves.



hashes and databases

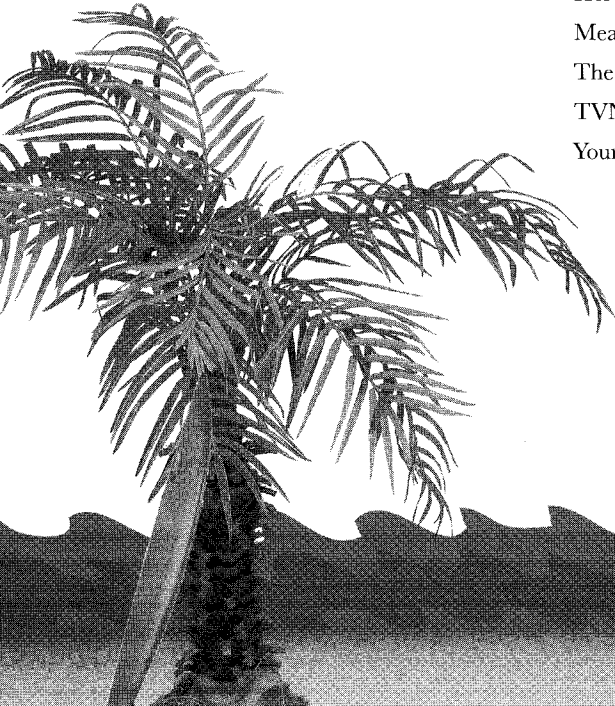
5

Putting data in its place

Arrays aren't the only show in town when it comes to data.

Programming languages come with other data-arranging goodies too, and our chosen tool, Python, is no exception. In this chapter, you'll **associate** values with names using a data structure commonly called the **hash** (better known as *dictionary* to Python-folk). And when it comes to working with **stored data**, you'll read data from an *external database system* as well as from regular text-based files. All the world's awash with data, so turn the page and start applying your ever-expanding programming skills to some cool data-processing tasks.

Who won the surfing contest?	146
Associate the name with the score	150
Associate a key with a value using a hash	153
Iterate hash data with for	154
The data isn't sorted	158
When data gets complex	160
Return a data structure from a function	164
Here's your new board!	168
Meanwhile, down at the studio...	169
The code remains the same; it's the function that changes	170
TVN's data is on the money!	174
Your Programming Toolbox	175

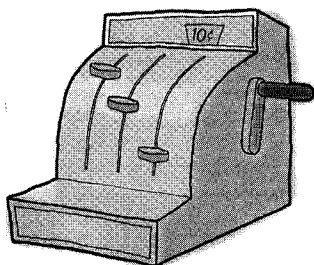
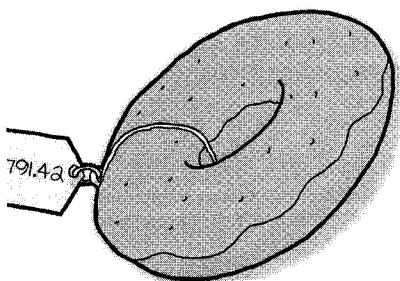


modular programming

Keeping things straight**6****The code that you write will make its way into many programs.**

And, although **sharing** is good, you need to be *careful*. One programmer might take your code and use it in an **unexpected** way, while another might change it without even letting you know. You might want to use one function in all your programs and, over time, that function's code might **change** to suit your needs. Smart programmers take advantage of *modular programming techniques* to keep their workload manageable.

Head First Health Club is upgrading some systems	178
The program needs to create a transaction file	179
Use strings to format strings	180
The Format String Exposed	186
A late night email ruins your day	187
\$50,000... for a donut?!	188
Only the sales from your program were rejected	189
The new bank uses a new format	190
Your coffee bar program still uses the old format	191
Don't just update your copy	192
So how do you create a module...?	193
The transaction file is working great, too	199
The health club has a new requirement	200
The Starbuzz code	205
The two discount functions have the same name	206
Fully Qualified Names (FQNs) prevent your programs from getting confused	207
The discounts get the customers flooding in	213
Your Programming Toolbox	214



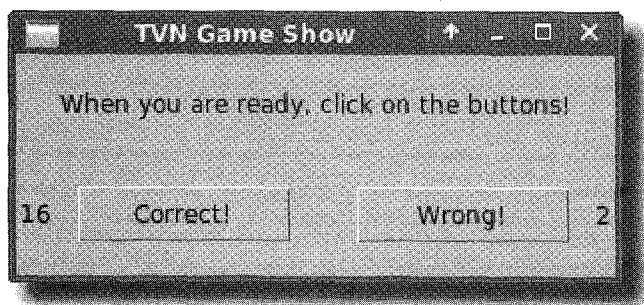
building a graphical user interface

7

Going all gooey**Your coding skills are great and getting better all the time.**

It's just a shame your programs are not that *nice* to look at. Displaying prompts and messages on a text-based console is all well and good, but it's so 1970s, isn't it? Add some green text on a black background and your retro look will be complete. There has to be a *better way* to communicate with your users than the console, and there is: using a **graphical user interface** or **GUI** (pronounced "gooey"). Sounds cool, but complex, and it can be. But, don't fret; learning a trick or two will have your code all graphical in no time. Let's get all gooey (sorry, GUI) in this chapter.

Head First TVN now produces game shows	216
pygame is cross platform	220
pygame Exposed	229
0... 2... 1... 9... blast off!	230
tkinter gives you the event loop for free	234
tkinter is packed with options	235
The GUI works, but doesn't do anything	238
Connect code to your button events	239
The GUI program's now ready for a screentest	244
But TVN is still not happy	246
Label it	249
Your Programming Toolbox	255



guis and data

8

Data entry widgets**GUIs don't just process events. They also handle data.**

Almost all GUI applications need to read user data, and choosing the right widgets can change your interface from *data entry hell* to *user heaven*. Widgets can accept plain text, or just present a menu of options. There are lots of different widgets out there, which means there are lots of choices, too. And, of course, making the right choice can make all the difference. It's time to take your GUI program to the **next level**.

Head-Ex needs a new delivery system	258
They've already designed the interface	259
Read data from the GUI	260
The Entry and Text widgets let you enter text data into your GUI	261
Read and write data to text fields	262
Large Text fields are harder to handle	263
One of the Head-Ex deliveries went astray	270
Users can enter anything in the fields	271
Radio buttons force users to choose a valid depot	272
Creating radio buttons in tkinter	273
The radio buttons should work together	275
The radio buttons can share a model	276
The system tells the other widgets when the model changes	277
So how do you use models in tkinter?	278
Head-Ex's business is expanding	282
There are too many depots on the GUI	283
An OptionMenu lets you have as many options as needed	284
The model stays the same	285
Things are going great at Head-Ex	291
Your Programming Toolbox	292

Look, I don't care
what you guys do, I'm
gonna stay selected.



Cambridge, MA

Yeah,
me too.



Cambridge, UK

Huh, and me.



Seattle, WA

exceptions and message boxes

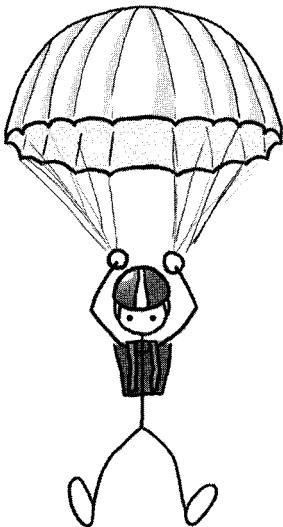
8½

Get the message?

Sometimes things just go wrong. You just need to handle it.

There will always be things beyond your control. Networks will fail. Files will disappear. Smart coders learn how to deal with those kinds of **errors** and make their programs **recover** gracefully. The best software keeps the user informed about the bad things that happen and what should be done to recover. By learning how to use **exceptions** and **message boxes**, you can take your software to the next level of reliability and quality.

What's that smell?	294
Someone changed the file permissions	295
When it couldn't write to the file, the program threw an exception	296
Catch the exception	297
Watch for exceptions with try/except	298
There's an issue with the exception handler	302
A message box demands attention	303
Creating message boxes in Python	304
Your Programming Toolbox	311



graphical interface elements

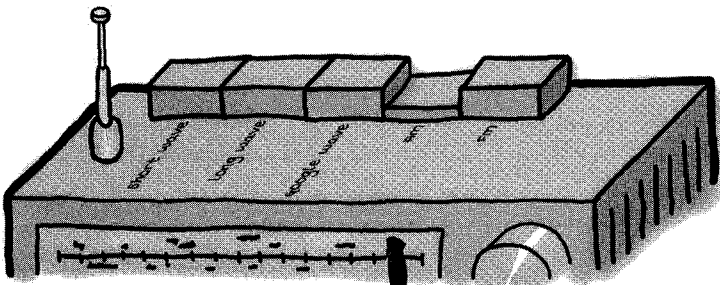
Selecting the right tool

9

It's easy to make your programs more effective for your users.

And when it comes to GUI applications, there's a world of difference between a *working* interface and one that's both **useful** and **effective**. Selecting the right tool for the right job is a skill that comes with experience, and the best way to get that experience is to use the tools available to you. In this chapter, you'll continue to expand your GUI application building skills. There's a bunch of truly useful widgets waiting to be experienced. So, turn the page and let's get going.

Time to mix it up	314
The music just kept on playing..	318
Not all events are generated by button clicks	319
Capturing the protocol event isn't enough	326
Two buttons, or not two buttons? That is the question...	328
The checkbox is an on/off, flip/flop toggle	331
Working with checkboxes in tkinter	332
Pump up the volume!	336
Model a slider on a scale	337
Use pygame to set the volume	339
Use tkinter for everything else	340
The DJ is over the moon!	347
Your Programming Toolbox	348



10

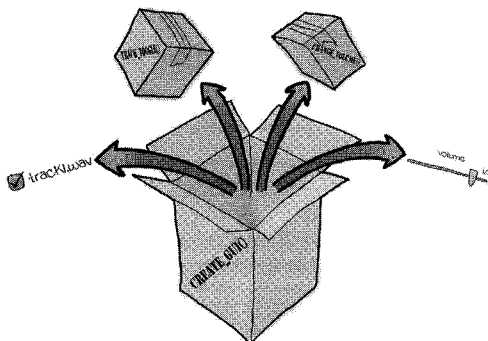
custom widgets and classes

With an object in mind

Requirements can be complex, but programs don't have to be.

By using object orientation, you can give your programs **great power** without writing lots of extra code. Keep reading, and you'll create **custom widgets** that do exactly what *you* want and give you the power to take **your programming skills to the next level**.

The DJ wants to play more than one track	350
Create code for each track as a function	351
The new function contains other functions	356
Your new function needs to create widgets and event handlers	357
The DJ is confused	362
Group widgets together	363
A frame widget contains other widgets	364
A class is a machine for creating objects	366
A class has methods that define behavior	367
But how does an object call a method?	369
The SoundPanel class looks a lot like the <code>create_gui()</code> function	370
class = methods + data	372
The Class Exposed	373
The DJ has an entire directory of tracks	378
It's party time!	382
Your Programming Toolbox	383
Leaving town...	384
It's been great having you here in Codeville!	384



leftovers

The Top Ten Things (we didn't cover)**You've come a long way.**

But learning how to program is an activity that never stops. The more you code, the more you'll need to **learn new ways to do certain things**. You'll need to master **new tools** and **new techniques**, too. There's just not enough room in this book to show you everything you might possibly need to know. So, here's our list of the top ten things we didn't cover that you might want to learn more about next.

#1: Doing things “The Python Way”	386
#2: Using Python 2	387
#3: Other programming languages	388
#4: Automated testing techniques	389
#5: Debugging	390
#6: Command-line execution	391
#7: Ooops... we could've covered more OOP	392
#8: Algorithms	393
#9: Advanced programming topics	394
#10: Other IDEs, shells and text editors	395

