LEARNING MADE EASY

2nd Edition

Python®

ALL-IN-ONE

for dummies®
A Wiley Brand

7 Books in one!

John C. Shovic, PhD
Alan Simpson

Python®

ALL-IN-ONE

2nd Edition

by John C. Shovic, PhD
Alan Simpson

for dummies®
A Wiley Brand

Python® All-in-One For Dummies®, 2nd Edition

Published by: **John Wiley & Sons, Inc.,** 111 River Street, Hoboken, NJ 07030-5774, www.wiley.com

Published simultaneously in Canada

PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For general information on our other products and services, please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993, or fax 317-572-4002. For technical support, please visit https://hub.wiley.com/community/support/dummies.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this

material at http://booksupport.wiley.com. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2021932818

ISBN 978-1-119-78760-0 (pbk); ISBN 978-1-119-78761-7 (ebk); ISBN 978-1-119-78762-4 (ebk)

Python® All-in-One For Dummies®

To view this book's Cheat Sheet, simply go to www.dummies.com and search for “Python All-in-One For Dummies Cheat Sheet” in the Search box.

Table of Contents

- Cover
- Title Page
- Copyright
- Introduction
  - About This Book
  - Foolish Assumptions
  - What to Buy
  - Icons Used in This Book
  - Beyond the Book
  - Where to Go from Here
- Book 1: Getting Started
  - Chapter 1: Starting with Python
    - Why Python Is Hot
    - Choosing the Right Python
    - Tools for Success
    - Writing Python in VS Code

Using Jupyter Notebook for Coding

Chapter 2: Interactive Mode, Getting Help, and Writing Apps

Using Python's Interactive Mode

Creating a Python Development Workspace

Creating a Folder for Your Python Code

Typing, Editing, and Debugging Python Code

Writing Code in a Jupyter Notebook

Chapter 3: Python Elements and Syntax

The Zen of Python

Introducing Object-Oriented Programming

Discovering Why Indentations Count, Big Time

Using Python Modules

Chapter 4: Building Your First Python Application

Opening the Python App File

Typing and Using Python Comments

Understanding Python Data Types

Working with Python Operators

Creating and Using Variables

Understanding What Syntax Is and Why It Matters

Putting Code Together

Book 2: Understanding Python Building Blocks

Chapter 1: Working with Numbers, Text, and Dates

Calculating Numbers with Functions

Still More Math Functions

Formatting Numbers

Grappling with Weirder Numbers

Manipulating Strings

Uncovering Dates and Times

Accounting for Time Zones

Working with Time Zones

Chapter 2: Controlling the Action

Main Operators for Controlling the Action

Making Decisions with if

Repeating a Process with for

Looping with while

Chapter 3: Speeding Along with Lists and Tuples

Defining and Using Lists

What's a Tuple and Who Cares?

Working with Sets

Chapter 4: Cruising Massive Data with Dictionaries

Understanding Data Dictionaries

Creating a Data Dictionary

Looping through a Dictionary

Data Dictionary Methods

Copying a Dictionary

Deleting Dictionary Items

Having Fun with Multi-Key Dictionaries

Chapter 5: Wrangling Bigger Chunks of Code

Creating a Function

Commenting a Function

Passing Information to a Function

Returning Values from Functions

Unmasking Anonymous Functions

Chapter 6: Doing Python with Class

Mastering Classes and Objects

Creating a Class

Creating an Instance from a Class

Giving an Object Its Attributes

Giving a Class Methods
Understanding Class Inheritance
Chapter 7: Sidestepping Errors
Understanding Exceptions
Handling Errors Gracefully
Being Specific about Exceptions
Keeping Your App from Crashing
Adding an else to the Mix
Using try ... except ... else ... finally
Raising Your Own Exceptions
Book 3: Working with Libraries
Chapter 1: Working with External Files
Understanding Text and Binary Files
Opening and Closing Files
Reading a File's Contents
Looping through a File
Reading and Copying a Binary File
Conquering CSV Files
Converting from CSV to Objects and Dictionaries
Chapter 2: Juggling JSON Data
Organizing JSON Data
Understanding Serialization
Loading Data from JSON Files
Dumping Python Data to JSON
Chapter 3: Interacting with the Internet
Seeing How the Web Works
Chapter 4: Libraries, Packages, and Modules
Understanding the Python Standard Library
Exploring Python Packages
Importing Python Modules
Making Your Own Modules

Book 4: Using Artificial Intelligence

Chapter 1: Exploring Artificial Intelligence

AI Is a Collection of Techniques

Current Limitations of AI

Chapter 2: Building a Neural Network

Understanding Neural Networks

Building a Simple Neural Network in Python

Building a Python Neural Network in TensorFlow

Chapter 3: Doing Machine Learning

Learning by Looking for Solutions in All the Wrong Places

Creating a Machine-Learning Network for Detecting Clothes Types

Visualizing with MatPlotLib

Learning More Machine Learning

Chapter 4: Exploring AI

Limitations of the Raspberry Pi and AI

Adding Hardware AI to the Raspberry Pi

AI in the Cloud

AI on a Graphics Card

Where to Go for More AI Fun in Python

Book 5: Doing Data Science

Chapter 1: Understanding the Five Areas of Data Science

Working with Big, Big Data

Cooking with Gas: The Five-Step Process of Data Science

Chapter 2: Exploring Big Data

Introducing NumPy, Pandas, and MatPlotLib

Doing Your First Data Science Project

Chapter 3: Using Big Data from Google Cloud

What Is Big Data?

Understanding Google Cloud and BigQuery
Reading the Medicare Big Data
Looking for the Most Polluted City in the World on an Hourly Basis

Book 6: Talking to Hardware

Chapter 1: Introducing Physical Computing

Physical Computing Is Fun
What Is a Raspberry Pi?
Building Projects That Move and Sense the Environment
Sensing the Environment with the Raspberry Pi
Controlling an LED with Python
But Wait, There's More

Chapter 2: No Soldering! Using Grove Connectors for Building

Working with the Grove System
Grove Connectors
Connecting with Grove Cables

Chapter 3: Sensing the World

Understanding I2C
Measuring Oxygen and a Flame
Building a Dashboard on Your Phone with Blynk
Where to Go from Here

Chapter 4: Making Things Move

Exploring Electric Motors
Controlling a DC Motor
Running a Servo Motor
Making a Stepper Motor Step

Book 7: Building Robots

Chapter 1: Introducing Robotics

A Robot Is Not Always Like a Human
Not Every Robot Has Arms or Wheels
Understanding the Main Parts of a Robot

Programming Robots

Chapter 2: Building Your First Python Robot

Introducing the Mars Rover PiCar-B

Assembling the Robot

Testing Your Robot

Chapter 3: Programming Your Robot Rover

Building a Simple, High-Level Python Interface

Making a Single Move with Python

Functions of the RobotInterface Class

The Python Robot Interface Test

Coordinating Motor Movements with Sensors

Making a Python Brain for Our Robot

Overview of the Included Adeept Software

Where to Go from Here

Chapter 4: Using Artificial Intelligence in Robotics

This Chapter’s Projects: Going to the Dogs

Setting Up the First Project

Machine Learning Using TensorFlow

Testing the Trained Network

Taking Cats and Dogs to Our Robot

Setting Up the Second Project

The FindAndChaseTheBall.py Python Program

The Main Program

AI and the Future of Robotics

Index

About the Authors

Connect with Dummies

End User License Agreement

List of Tables

Book 1 Chapter 1

TABLE 1-1 Examples of Python Versions and Release Dates

Book 1 Chapter 4

TABLE 4-1 Examples of Good and Bad Python Numbers

TABLE 4-2 Python’s Arithmetic Operators

TABLE 4-3 Python Comparison Operators

TABLE 4-4 Python Boolean Operators

Book 2 Chapter 1

TABLE 1-1 Some Built-In Python Functions for Numbers

TABLE 1-2 Some Functions from the Python Math Module

TABLE 1-3 Python for Base 2, 8, and 16 Numbers

TABLE 1-4 Python Sequence Operators That Work with Strings

TABLE 1-5 Built-In Methods for Python 3 Strings

TABLE 1-6 Formatting Strings for Dates and Times

TABLE 1-7 Sample Date Format Strings

TABLE 1-8 Sample Date Format Strings

TABLE 1-9 Sample Datetime Format Strings

TABLE 1-10 Sample Time Zones from the Olson Database

Book 2 Chapter 2

TABLE 2-1 Python Comparison Operators for Decision-Making

TABLE 2-2 Python Logical Operators

Book 2 Chapter 3

TABLE 3-1 Methods for Working with Lists

Book 2 Chapter 4

TABLE 4-1 Data Dictionary Methods

TABLE 4-2 A Table of Products

Book 3 Chapter 2

TABLE 2-1 Python JSON Methods for Serializing and Deserializing JSON Data

TABLE 2-2 Python and JSON Data Conversions

Book 3 Chapter 3

TABLE 3-1 Common HTTP Status Codes

TABLE 3-2 Packages from the Python `urllib` Library

Book 4 Chapter 2

TABLE 2-1 The Truth Table (a Three-Input XNOR Gate) for the Neural Network

Book 5 Chapter 2

TABLE 2-1 Columns in the Diamond Database

Book 5 Chapter 3

TABLE 3-1 Columns, Types, and Descriptions of the inpatient_charges_2015 Dataset

Book 6 Chapter 2

TABLE 2-1 The Grove Digital Connector

TABLE 2-2 The Grove Analog Connector

TABLE 2-3 The Grove UART Serial Connector

TABLE 2-4 The Grove I2C Connector

Book 6 Chapter 4

TABLE 4-1 Servo Motor to Patch Cable Wiring

TABLE 4-2 Forward Stepping the Stepper

TABLE 4-3 Backward Stepping the Stepper

TABLE 4-4 First Grove Female Patch Cord to UNL2003 Driver Board

TABLE 4-5 Second Grove Female Patch Cord to UNL2003 Driver Board

List of Illustrations

Book 1 Chapter 1

FIGURE 1-1: Google search trends for the last five years or so.

FIGURE 1-2: Click Download under the largest version number.

FIGURE 1-3: In Windows, right-click and choose Run As Administrator.

FIGURE 1-4: Choose how to install Anaconda.

FIGURE 1-5: Anaconda Navigator home page.

FIGURE 1-6: The welcome screen of VS Code editor.

FIGURE 1-7: VS Code extensions for Python.

FIGURE 1-8: Choose your Python interpreter (usually the highest version number)...

FIGURE 1-9: Terminal in VS Code (Windows and Mac).

FIGURE 1-10: Python shows the sum of one plus one.

FIGURE 1-11: Launch Jupyter Notebook from Anaconda’s home page.

FIGURE 1-12: Jupyter Notebook opening page.

FIGURE 1-13: Creating a new Jupyter notebook.

FIGURE 1-14: Two ways to run code in a Jupyter cell.

FIGURE 1-15: Result of running code in a Jupyter Notebook cell.

Book 1 Chapter 2

FIGURE 2-1: The Terminal pane in VS Code.

FIGURE 2-2: Python doesn't know what *howdy* means.

FIGURE 2-3: Python’s interactive help utility.

FIGURE 2-4: Keyword help.

FIGURE 2-5: Python class help.

FIGURE 2-6: Back to the operating system prompt.

FIGURE 2-7: Saving current settings as workspace settings.

FIGURE 2-8: VS Code Settings.

FIGURE 2-9: Python path copied to Workspace Settings.

FIGURE 2-10: Python 3 workspace and AIO Python folder open in VS Code.

FIGURE 2-11: Right-click a folder name and choose New File.

FIGURE 2-12: New `hello.py` file is open for editing in VS Code.

FIGURE 2-13: The `hello.py` file contains some Python code and has unsaved change...

FIGURE 2-14: Run `hello.py`.

FIGURE 2-15: Output from `hello.py`.

FIGURE 2-16: `PRINT` is typed incorrectly in `hello.py`.

FIGURE 2-17: VS Code Debug pane.

FIGURE 2-18: A saved Jupyter notebook.

FIGURE 2-19: A Markdown cell containing some Markdown content.

FIGURE 2-20: A Markdown cell with some Markdown code and text in it.

Book 1 Chapter 3

FIGURE 3-1: The Zen of Python.

FIGURE 3-2: Workspace settings with PyLint and Pycodestyle (PEP 8) enabled.

FIGURE 3-3: A different view of Workspace settings.

FIGURE 3-4: Installed modules.

FIGURE 3-5: All our packages are installed and up-to-date.

Book 1 Chapter 4

FIGURE 4-1: The `hello.py` file, open for editing in VS Code.

FIGURE 4-2: A comment in `hello.py`.

FIGURE 4-3: Your name and the date you became a customer appear on Amazon's hom...

FIGURE 4-4: Your first Python app typed into VS Code.

FIGURE 4-5: Right-click a `.py` file and choose Run Python File in Terminal.

FIGURE 4-6: The 19.9 is the output from `print(extended_price)` in the code.

FIGURE 4-7: Touching the mouse pointer to a red wavy underline.

FIGURE 4-8: Touching the mouse pointer to a green wavy underline.

Book 2 Chapter 1

FIGURE 1-1: Trying out the `abs()` function.

FIGURE 1-2: Trying out the `round()` function.

FIGURE 1-3: Playing around with built-in math functions at the Python prompt.

FIGURE 1-4: Using the `sqrt()` function from the `math` module.

FIGURE 1-5: More playing around with built-in math functions at the Python prom...

FIGURE 1-6: A super simple f-string for formatting.

FIGURE 1-7: Formatting a percentage number.

FIGURE 1-8: An f-string can be encased in single, double, or triple quotation m...

FIGURE 1-9: A multiline f-string enclosed in triple quotation marks.

FIGURE 1-10: All dollar amounts are right aligned within a width of 9 character...

FIGURE 1-11: All the dollar amounts neatly aligned.

FIGURE 1-12: Messing about with binary, octal, and hex.

FIGURE 1-13: Character positions in a string start at 0, not 1.

FIGURE 1-14: Playing around with string operators in Jupyter Notebook.

FIGURE 1-15: ASCII numbers for common characters.

FIGURE 1-16: Playing around with Python 3 string functions.

FIGURE 1-17: Experiments with `datetime.date` objects in a Jupyter notebook.

FIGURE 1-18: Calculating age in years and months from a `timedelta` object.

FIGURE 1-19: Time zones.

FIGURE 1-20: Determining the difference between your time and UTC time.

FIGURE 1-21: The current date and time for five different time zones.

FIGURE 1-22: Date and time for a scheduled event in multiple time zones.

Book 2 Chapter 2

FIGURE 2-1: The result of a simple `if` when the condition proves true.

FIGURE 2-2: Result of simple `if` when the condition proves false.

FIGURE 2-3: Result of simple `if` when the condition proves true and then false.

FIGURE 2-4: When `taxable` is `True`, `sales_tax` is added to the total.

FIGURE 2-5: When `taxable` is `False`, `sales_tax` is not added into the total.

FIGURE 2-6: Print an initial greeting based on the time of day.

FIGURE 2-7: A loop that counts from 1 to 10.

FIGURE 2-8: Looping through a list.

FIGURE 2-9: Looping through a list.

FIGURE 2-10: Nested loops.

FIGURE 2-11: Looping while `counter` is less than 91.

FIGURE 2-12: An infinite `while` loop.

FIGURE 2-13: A `while` loop with `continue`.

FIGURE 2-14: A `while` loop with `break`.

FIGURE 2-15: The same code as in Figure 2-14 on a second run.

Book 2 Chapter 3

FIGURE 3-1: Index out-of-range error because `scores[5]` doesn't exist.

FIGURE 3-2: Looping through a list.

FIGURE 3-3: Seeing whether an item is in a list.

FIGURE 3-4: Appending two new names to the end of the list.

FIGURE 3-5: Removing list items with `pop()`.

FIGURE 3-6: Deleting a list and then trying to print it causes an error.

FIGURE 3-7: Counting items in a list.

FIGURE 3-8: Program fails when trying to find the index of a nonexistent list i...

FIGURE 3-9: Sorting strings and numbers.

FIGURE 3-10: Sorting and displaying dates in a nice format.

FIGURE 3-11: Sorting strings, numbers, and dates in reverse order.

FIGURE 3-12: Playing about with Python sets.

Book 2 Chapter 4

FIGURE 4-1: A data dictionary with keys in the left column and values in the ri...

FIGURE 4-2: A data dictionary with lists as values.

FIGURE 4-3: A data dictionary for one employee.

FIGURE 4-4: A data dictionary for another employee.

FIGURE 4-5: Printing the value of the `zmin` key in the `people` dictionary.

FIGURE 4-6: Python's way of saying there is no *schmeedledorp*.

FIGURE 4-7: Seeing if a key exists in a dictionary.

FIGURE 4-8: Python's nicer way of saying there is no *schmeedledorp.*

FIGURE 4-9: Changing the value associated with a key in a dictionary.

FIGURE 4-10: Looping through a dictionary with `items()` and two variable names.

FIGURE 4-11: Copying a dictionary.

FIGURE 4-12: Popping an item from a dictionary.

FIGURE 4-13: Experimenting with `fromkeys` and `setdefault`.

FIGURE 4-14: Multiple product dictionaries contained in a larger `products` dicti...

FIGURE 4-15: Printing data dictionaries formatted into rows and columns.

Book 2 Chapter 5

FIGURE 5-1: Writing, and calling, a simple function named `hello()`.

FIGURE 5-2: The docstring comment for your function appears in VS Code IntelliS...

FIGURE 5-3: Passing data to a function via a variable.

FIGURE 5-4: An optional parameter with a default value added to the `hello()` fun...

FIGURE 5-5: The `hello` function with three parameters.

FIGURE 5-6: Calling the `hello()` function with three parameters, and again with ...

FIGURE 5-7: Calling a function with keyword arguments (kwargs).

FIGURE 5-8: Using the `alphabetize` function in VS Code.

FIGURE 5-9: A function accepting any number of arguments with `*args`.

FIGURE 5-10: Printing a string returned by the `alphabetize()` function.

FIGURE 5-11: Putting a custom function named `lowercaseof()` to the test.

FIGURE 5-12: Using a lambda expression as a sort key.

FIGURE 5-13: Two anonymous functions for formatting numbers.

FIGURE 5-14: Two functions for formatting numbers with a fixed width.

Book 2 Chapter 6

FIGURE 6-1: Different car objects.

FIGURE 6-2: The Dog class creates many unique dogs.

FIGURE 6-3: The `Member` class and member instances.

FIGURE 6-4: Creating a member from the `Member` class in a Jupyter cell.

FIGURE 6-5: The `Member` class with `username` and `fullname` for both parameters and...

FIGURE 6-6: VS Code displays help when you access your own custom classes.

FIGURE 6-7: Changing the value of an object's attribute.

FIGURE 6-8: Changing the value of an object's attributes.

FIGURE 6-9 Adding and testing an `.activate()` method.

FIGURE 6-10: The `free_days` variable is a class variable in the `Member` class.

FIGURE 6-11: The `setfreedays()` method is a class method in the `Member` class.

FIGURE 6-12: The `Member` class now has a static method named `currenttime()`.

FIGURE 6-13: Dogs as objects of the class dogs.

FIGURE 6-14: Several different kinds of animals are similar to dogs.

FIGURE 6-15: A simplified `Member` class.

FIGURE 6-16: Creating and testing the `Admin` and `User` classes.

FIGURE 6-17: The `Admin` subclass has a new `secret_code` parameter.

FIGURE 6-18: The complete `Admin` and `User` subclasses.

FIGURE 6-19: Three methods with the same name, `get_status()`.

FIGURE 6-20: Output from `help(Admin)`.

Book 2 Chapter 7

FIGURE 7-1: The `showfilecontents.py` and `people.csv` files in a folder in VS Code...

FIGURE 7-2: The contents of the `people.csv` file in Excel (top) and a text edito...

FIGURE 7-3: The `showfilecontents.py` file raises an exception.

FIGURE 7-4: The `showfilecontents.py` file catches the error and displays a frien...

FIGURE 7-5: The correct error message is displayed.

FIGURE 7-6: Code with try, exception handlers, and an else for when there are n...

FIGURE 7-7: Custom `EmptyFileError` exception added for exception handling.

Book 3 Chapter 1

FIGURE 1-1: How a binary files looks in a program for editing text files.

FIGURE 1-2: Common text and binary files.

FIGURE 1-3: How `happy_pickle.jpg` is supposed to look.

FIGURE 1-4: The `Names.txt` file is text, but with lots of non-English characters...

FIGURE 1-5: Contents of `names.txt` displayed.

FIGURE 1-6: A new name appended to the end of the `names.txt` file.

FIGURE 1-7: The `binarycopy.py` file copies any binary file.

FIGURE 1-8: Running `binarycopy.py` added `happy_pickle_copy.jpg` to the folder.

FIGURE 1-9: A CSV file in Microsoft Excel.

FIGURE 1-10: A CSV file in a text editor.

FIGURE 1-11: Reading a CSV file and converting it to Python data types.

FIGURE 1-12: Reading a CSV file into a list of objects.

FIGURE 1-13: Reading a CSV file into a dictionary of dictionaries.

Book 3 Chapter 2

FIGURE 2-1: Some data in an Excel spreadsheet.

FIGURE 2-2: Excel spreadsheet data converted to JSON format.

FIGURE 2-3: Some data in a Google Firebase Realtime Database.

FIGURE 2-4: Google Firebase Realtime Database data exported to a keyed JSON fil...

FIGURE 2-5: Output from looping through and displaying keys and values from sub...

FIGURE 2-6: Output showing one value at a time from each dictionary.

FIGURE 2-7: Output from showing one value at a time from each dictionary (see b...

FIGURE 2-8: Changing the value of one key in each dictionary, and removing an e...

FIGURE 2-9: Writing modified Firebase data to a new JSON file named `hitcounts_n`...

FIGURE 2-10: Writing modified Firebase data to a new JSON file named `hitcounts_`...

Book 3 Chapter 3

FIGURE 3-1: The client makes a request, and the server sends back a response.

FIGURE 3-2: Different parts of URLs.

FIGURE 3-3: Inspecting HTTP headers with Google Chrome.

FIGURE 3-4: HTTP headers.

FIGURE 3-5: Sample page used for web scraping.

FIGURE 3-6: Some of the code from the sample page for web scraping.

FIGURE 3-7: Web scraping code complete.

FIGURE 3-8: Web scraped data in a JSON file.

FIGURE 3-9: Web scraped data in Excel.

FIGURE 3-10: The entire `scraper.py` program.

Book 3 Chapter 4

FIGURE 4-1: Python's built-in functions.

FIGURE 4-2: Installed packages as viewed in Anaconda.

Book 4 Chapter 2

FIGURE 2-1: A two-layer neural network.

FIGURE 2-2: Feed-forward and backpropagation.

FIGURE 2-3: An example of a sigmoid function.

FIGURE 2-4: The `Loss` function during training.

FIGURE 2-5: Our TensorFlow three-layer neural network.

FIGURE 2-6: Results of the two-layer training.

FIGURE 2-7: Results of the three-layer training.

Book 4 Chapter 3

FIGURE 3-1: A small portion of the Fashion-MNIST database.

FIGURE 3-2: A full GUI on the Raspberry Pi.

FIGURE 3-3: Image 15 from the Fashion-MNIST test database.

FIGURE 3-4: Unclassified dress hanging on a wall.

FIGURE 3-5: The dress at 28x28 pixels.

FIGURE 3-6: Our Raspberry Pi GUI with MatPlotLib visualization.

Book 4 Chapter 4

FIGURE 4-1: The Raspberry Pi processing chip containing the Videocore-IV.

FIGURE 4-2: The Intel Neural Compute Stick 2.

FIGURE 4-3: The Google Edge TPU accelerator.

FIGURE 4-4: Nvidia 256 Core GPU chip.

Book 5 Chapter 2

FIGURE 2-1: Diamond clarity (horizontal) versus carat size (vertical).

FIGURE 2-2: Diamond clarity count in each type.

FIGURE 2-3: Diamond color count in each type.

FIGURE 2-4: Correlation heat chart.

Book 5 Chapter 3

FIGURE 3-1: Google Cloud developer's console page.

FIGURE 3-2: First credentials screen.

FIGURE 3-3: Second credentials screen.

FIGURE 3-4: Bar chart of Medicare percent paid per state for code 554.

Book 6 Chapter 1

FIGURE 1-1: The main components of the Raspberry Pi 3B+.

FIGURE 1-2: The functions of the Raspberry Pi GPIO pins.

FIGURE 1-3: The Pi2Grover board.

FIGURE 1-4: The Grove blue LED.

FIGURE 1-5: Aligning the Pi2Grover board with the Raspberry Pi.

FIGURE 1-6: The installed Pi2Grover board.

FIGURE 1-7: A Grove cable plugged into the Grove blue LED board.

FIGURE 1-8: The LED aligned with the outline on the board.

FIGURE 1-9: The completed “Hello World” project.

FIGURE 1-10: Duty cycles.

Book 6 Chapter 2

FIGURE 2-1: The Arduino Uno Grove base board.

FIGURE 2-2: The Arduino Mini Pro LB board with Grove.

FIGURE 2-3: The Pi2Grover board at work on the Raspberry Pi.

FIGURE 2-4: A Grove connector.

FIGURE 2-5: 5cm-long Grove cables.

FIGURE 2-6: A simple digital Grove module with LED.

FIGURE 2-7: A Grove analog simple voltage divider.

FIGURE 2-8: A Grove UART RFID reader.

FIGURE 2-9: The Grove I2C sunlight sensor.

FIGURE 2-10: 20cm Grove cables.

FIGURE 2-11: Grove female header cables.

FIGURE 2-12: Grove male header cables.

FIGURE 2-13: The SunAirPlus board with the Grove female header patch cable.

FIGURE 2-14: A Grove adaptor cable attached to Pi2Grover.

FIGURE 2-15: A close-up of the Adafruit GPS with a Grove patch cable.

Book 6 Chapter 3

FIGURE 3-1: The I2C bus.

FIGURE 3-2: HDC1080 temperature and humidity sensor.

FIGURE 3-3: HDC1080 with the Grove cable plugged in.

FIGURE 3-4: The HDC1080 hooked up to the Raspberry Pi.

FIGURE 3-5: The Grove four-channel, 16-bit ADC.

FIGURE 3-6: The Grove oxygen sensor.

FIGURE 3-7: The complete Raspberry Pi/ADC/oxygen sensor hookup.

FIGURE 3-8: The start of our O2 experiment.

FIGURE 3-9: The graph of the data from our O2 experiment.

FIGURE 3-10: The MyTemperature dashboard.

FIGURE 3-11: Blynk in the App Store (left) and creating a Blynk account (right)...

FIGURE 3-12: Click for QR (left), and then scan the QR to generate your myTempe...

FIGURE 3-13: The MyTemperature app (left) and the initial screen of the Blynk a...

FIGURE 3-14: The authentication token in the MyTemperature app project settings...

FIGURE 3-15: The MyTemperature app’s Live view.

Book 6 Chapter 4

FIGURE 4-1: A DC motor on a small robot.

FIGURE 4-2: Sun-tracking solar panels using a stepper motor.
FIGURE 4-3: The Grove I2C motor driver.
FIGURE 4-4: Annotated diagram of the I2C motor driver board.
FIGURE 4-5: The Adafruit DC motor.
FIGURE 4-6: The wires in the I2C motor drive screw terminals.
FIGURE 4-7: Motors installed on the motor drive.
FIGURE 4-8: The DC motor setup.
FIGURE 4-9: The SG90 micro servo with wires.
FIGURE 4-10: Grove male-pin-to Grove-connector patch cable.
FIGURE 4-11: Servo motor correctly wired to the patch cable.
FIGURE 4-12: Fully connected Pi and servo motor.
FIGURE 4-13: A diagram of a stepper motor.
FIGURE 4-14: Logic analyzer showing the motor stepping sequence.
FIGURE 4-15: The 28BYJ-48 stepper motor and UNL2003 driver board.
FIGURE 4-16: A Grove-connector-to-female-pin-header patch cable.
FIGURE 4-17: Closeup of power connections on the UNL2003 driver board.
FIGURE 4-18: Second Grove patch cable attached.
FIGURE 4-19: All patch wires installed on the UNL2003 driver board.
FIGURE 4-20: Stepper motor and driver board connected.
FIGURE 4-21: Fully wired Raspberry Pi and stepper motor project.
FIGURE 4-22: Stepper motor, ready to step.
FIGURE 4-23: The Raspberry Pi running the stepper motor.

Book 7 Chapter 1

FIGURE 1-1: Inputs for the BMW robot driving system.
FIGURE 1-2: A robot making bread.
FIGURE 1-3: Baxter making coffee.
FIGURE 1-4: The Toasteroid Internet-connected toaster.

Book 7 Chapter 2

FIGURE 2-1: The assembled PiCar-B robot.

FIGURE 2-2: The PiCar-B motor controller board.
FIGURE 2-3: The SG90 micro servo motor.
FIGURE 2-4: The main drive motor.
FIGURE 2-5: A single RGB LED.
FIGURE 2-6: The 12 programmable RGB LEDs.
FIGURE 2-7: Raspberry Pi camera and cable.
FIGURE 2-8: An ultrasonic distance sensor.
FIGURE 2-9: An example of the assembly manual diagrams.
FIGURE 2-10: The assembled PiCar-B showing wiring.
FIGURE 2-11: Setting the VNC viewer option.

Book 7 Chapter 3

FIGURE 3-1: Adeept remote control software.

Book 7 Chapter 4

FIGURE 4-1: Cats and dogs recognition accuracy per epoch.
FIGURE 4-2: Panther the cat on salmon.
FIGURE 4-3: Winston the dog.
FIGURE 4-4: A picture of a dress?
FIGURE 4-5: Robot vision neural network test setup.
FIGURE 4-6: The cat who is apparently a dog.
FIGURE 4-7: MouseAir, an AI mouse-launching cat toy.
FIGURE 4-8: The OpenCV processed frame.
FIGURE 4-9: The OpenCV mask frame.
FIGURE 4-10: Screen for the blue ball color configuration.
FIGURE 4-11: Raspberry Pi trying to chase the ball.
FIGURE 4-12: Missing balls for other targets.

Introduction

The power of Python. The Python language is becoming more and more popular, and in 2017 it became the most popular language in the world according to IEEE Spectrum. The power of Python is real.

Python the number-one language because it's easy to learn and use, due partly to its simplified syntax and natural-language flow but also to the amazing user community and the breadth of applications available.

About This Book

This book is a reference manual to guide you through the process of learning Python and how to use it in modern computer applications, such as data science, artificial intelligence, physical computing, and robotics. If you're looking to learn a little about a lot of exciting things, this is the book for you. It gives you an introduction to the topics that you'll need to explore more deeply.

Python All-in-One For Dummies, 2nd Edition guides you through the Python language and then takes you on a tour through some cool libraries and technologies (the Raspberry Pi, robotics, AI, data science, and more) that all revolve around the Python language. When you work on new projects and new technologies, Python is there with a diverse number of libraries just waiting for you to use.

This is a hands-on book, with examples and code throughout. You are expected to enter the code, run it, and then modify it to do what you want. You don’t just buy a robot; you build it so you can understand all the pieces and can make sense of the way Python works with

the robot to control its motors and sensors. Artificial intelligence is complicated, but Python helps make a significant part of it accessible. Data science is complicated, but Python helps you do data science more easily. Robotics is complicated, but Python gives you the code that controls the robot. And Python even enables you to tie these pieces together and use, say, AI in robotics.

In this book, we take you through the basics of the Python language in small, easy-to-understand steps. After we have introduced you to the language, we step into the world of artificial intelligence, exploring programming in machine learning and neural networks using Python and TensorFlow and working on real problems and real software, not just toy applications.

After that, we're off to the exciting world of big data and data science with Python. We look at big public data sets such as medical and environmental data.

Finally, you get to experience the magic of what we call physical computing. Using the inexpensive, small, and incredibly popular Raspberry Pi computer, we show you how to use Python to control motors and read sensors. This is a lead-up to the final minibook, "Building Robots," where you build a robot and control it with Python and your own programs, even using artificial intelligence. This is not your mother's RC car.

Python data science, robotics, AI, and fun all in the same book.

This book won't make you understand everything about these fields, but it will give you a great introduction to the terminology and the power of Python in all these fields. Enjoy the book and go forth and learn more afterwards.

Foolish Assumptions

We assume that you know how to use a computer in a basic way. If you can turn on the computer and use a mouse, you're ready for this book. We assume that you don't know how to program yet, although you will have some skills in programming by the end of the book. If we're wrong and you already know Python (or some other computer language), jump ahead to minibook 4 and dig right into learning something new. Our intent is to guide you through the language of Python and then through some of the amazing technologies and devices that use Python. We provide complete examples. If you get stuck on something, look it up on the web, read a tutorial, and then come back to it.

What to Buy

To complete the projects in Books 4 through 7, you need a Raspberry Pi 3B+ starter kit at https://amzn.to/2WzYdoY or a Raspberry PI 4B Starter Kit at https://amzn.to/3nIH8W8. In addition, you need the items listed in this section, organized by minibook.

TIP If you want to use a Raspberry Pi 4B in the robot in Book 7, it will dramatically reduce the battery life, and with some types of batteries the robot may not be able to boot the Pi 4B.

Book 6

For building the projects in Book 6, you need the following:

- Pi2Grover board at https://shop.switchdoc.com or www.amazon.com. (You can get $5.00 off the board at shop.switchdoc.com by using the discount code PI2DUMMIES at checkout.)
- Grove blue LED module, which includes a Grove cable, at https://shop.switchdoc.com or Amazon.
- A package of Grove male jumper patch cables, specifically the Grove-4-male-pin-to-Grove-conversion cables, at https://shop.switchdoc.com/products/grove-4-pin-male-jumper-to-grove-4-pin-conversion-cable-5-pcs-per-pack and https://amzn.to/3nyGbic.
- A package of female-to-Grove patch cables at https://shop.switchdoc.com/products/grove-4-pin-female-jumper-to-grove-4-pin-conversion-cable-5-pcs-per-pack and https://amzn.to/3jhQmXY.
- Grove HDC1080 I2C temperature and humidity sensor at https://store.switchdoc.com or www.amazon.com. The SwitchDoc Labs HDC1080 sensor comes with a Grove connector. If you buy a non-Grove sensor on Amazon, you'll need a female-to-Grove patch cable, as discussed in Chapter 2 of this minibook. You can get a female-to-Grove patch cable at https://shop.switchdoc.com/products/grove-4-pin-female-jumper-to-grove-4-pin-conversion-cable-5-pcs-per-pack and https://amzn.to/3jhQmXY.
- Grove oxygen sensor at www.seeedstudio.com or www.amazon.com.
- Pi2Grover Raspberry-Pi-to-Grove converter, https://shop.switchdoc.com or www.amazon.com. (You can get $5.00 off the board at shop.switchdoc.com by using the discount code PI2DUMMIES at checkout.)
- Grove four-channel, 16-bit analog-to-digital converter at https://store.switchdoc.com or www.amazon.com.