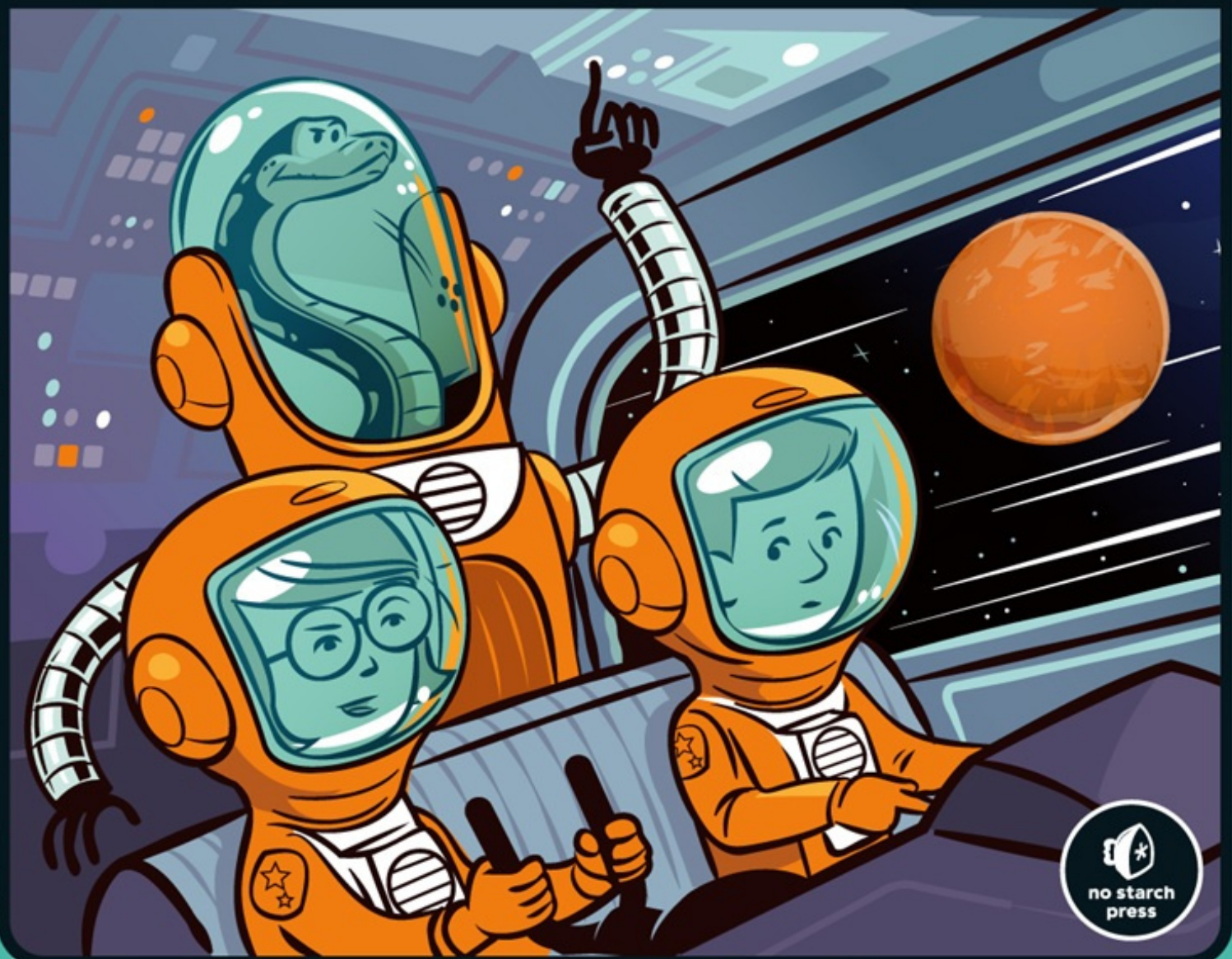


MISSION PYTHON

CODE A SPACE ADVENTURE GAME!

SEAN MCMANUS



MISSION PYTHON

CODE A SPACE ADVENTURE GAME!

BY SEAN MCMANUS



**no starch
press**

San Francisco

MISSION PYTHON. Copyright © 2018 by Sean McManus.

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-10: 1-59327-857-8

ISBN-13: 978-1-59327-857-1

Publisher: William Pollock

Production Editor: Riley Hoffman

Cover Illustration: Josh Ellingson

Game Illustrations: Rafael Pimenta

Developmental Editor: Liz Chadwick

Technical Reviewer: Daniel Aldred

Copyeditor: Anne Marie Walker

Compositor: Riley Hoffman

Proofreader: Emelie Burnette

The following images are reproduced with permission:

Figure 1-1 courtesy of Johnson Space Center, NASA

Figure 1-6 courtesy of NASA/JPL-Caltech/UCLA

Figure 1-7 image of Mars courtesy of NASA

For information on distribution, translations, or bulk sales, please contact No Starch Press, Inc. directly:

No Starch Press, Inc.

245 8th Street, San Francisco, CA 94103

phone: 1.415.863.9900; info@nostarch.com

www.nostarch.com

Library of Congress Control Number: 2018950581

No Starch Press and the No Starch Press logo are registered trademarks of No Starch Press, Inc. Other product and company names mentioned herein may be the trademarks of their respective owners. Rather than use a trademark symbol with every occurrence of a trademarked name, we are using the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The information in this book is distributed on an “As Is” basis, without warranty. While every precaution has been taken in the preparation of this work, neither the author nor No Starch Press, Inc. shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in it.

To my wife, Karen, with thanks for all her support throughout this mission;
and to Leo, our wonderful son, who is taking us on the most amazing
journey.

ABOUT THE AUTHOR

Sean McManus is an expert technology and business writer. His other books include *Cool Scratch Projects in Easy Steps*, *Scratch Programming in Easy Steps*, *Coder Academy*, and *Raspberry Pi For Dummies* (co-authored with Mike Cook). As a freelance copywriter, he writes for many of the world's largest technology companies. His novel for adults, *Earworm*, goes undercover in the music industry, exposing a conspiracy to replace bands with computer-generated music. He has been a Code Club volunteer, helping children at a local school to learn computer programming. Visit his website at www.sean.co.uk for sample chapters and bonus content from his books.

ABOUT THE TECHNICAL REVIEWER

Daniel Aldred is a passionate and experienced teacher of computer science. He leads the computing department at a CAS hub school that supports and develops other schools and organizations in the local area. He frequently writes for *Linux User & Developer* and has created resources and projects for Raspberry Pi, Pimoroni, micro:bit, and Cambridge International Assessment. In his spare time he curates two websites, www.canyoucompute.co.uk for the UK Computing examination course and www.tecoed.co.uk for his own personal hacks. Daniel also led and managed a team of eight students to win the first Astro Pi competition, where the astronaut Major Tim Peake ran their program aboard the ISS.

BRIEF CONTENTS

Acknowledgments

Introduction

Chapter 1: Your First Spacewalk

Chapter 2: Lists Can Save Your Life

Chapter 3: Repeat After Me

Chapter 4: Creating the Space Station

Chapter 5: Preparing the Space Station Equipment

Chapter 6: Installing the Space Station Equipment

Chapter 7: Moving into the Space Station

Chapter 8: Repairing the Space Station

Chapter 9: Unpacking Your Personal Items

Chapter 10: Make Yourself Useful

Chapter 11: Activating Safety Doors

Chapter 12: Danger! Danger! Adding Hazards

Appendix A: Escape: The Complete Game Listing

Appendix B: Table of Variables, Lists, and Dictionaries

Appendix C: Debugging Your Listings

Index

CONTENTS IN DETAIL

ACKNOWLEDGMENTS

INTRODUCTION

How to Use This Book

What's in This Book?

Installing the Software

- Installing the Software on Raspberry Pi

- Installing Python on Windows

- Installing Pygame Zero on Windows

- Installing the Software on Other Machines

Downloading the Game Files

- Downloading and Unzipping the Files on a Raspberry Pi

- Unzipping the File on a Windows PC

- What's in the ZIP File

Running the Game

- Running Pygame Zero Programs on the Raspberry Pi

- Running Pygame Zero Programs in Windows

Playing the Game

1

YOUR FIRST SPACEWALK

Starting the Python Editor

- Starting IDLE in Windows 10

- Starting IDLE in Windows 8

- Starting IDLE on the Raspberry Pi

Introducing the Python Shell

- Displaying Text

- Training Mission #1*

- Outputting and Using Numbers

Introducing Script Mode

Creating the Starfield

Understanding the Program So Far
Stopping Your Pygame Zero Program
Adding the Planet and Spaceship
 Changing Perspective: Flying Behind the Planet
 Training Mission #2
 Spacewalking!
 Training Mission #3
 Understanding the Spacewalk Listing
 Training Mission #4
Are You Fit to Fly?
 Mission Debrief

2

LISTS CAN SAVE YOUR LIFE

Making Your First List: The Take-Off Checklist
 Seeing Your List
 Adding and Removing Items
Using Index Numbers
 Inserting an Item
 Accessing an Individual Item
 Replacing an Item
 Deleting an Item
 Training Mission #1
Creating the Spacewalk Checklist
 Training Mission #2
A List of Lists: The Flight Manual
 Making a List of Lists
 Training Mission #3
 Finding an Item in the Flight Manual
Combining Lists
Making Maps from Lists: The Emergency Room
 Making the Map
 Finding an Emergency Item
 Training Mission #4

Swapping Items in the Room

Training Mission #5

Are You Fit to Fly?

Mission Debrief

3

REPEAT AFTER ME

Displaying Maps with Loops

Making the Room Map

Displaying the Map with a Loop

Training Mission #1

Loop the Loop

Nesting Loops to Get Room Coordinates

Cleaning Up the Map

Training Mission #2

Displaying a 3D Room Image

Understanding How the Room Is Drawn

Working Out Where to Draw Each Item

Training Mission #3

Are You Fit to Fly?

Mission Debrief

4

CREATING THE SPACE STATION

Automating the Map Making Process

How the Automatic Map Maker Works

Creating the Map Data

Writing the GAME_MAP Code

Testing and Debugging the Code

Generating Rooms from the Data

How the Room Generating Code Works

Creating the Basic Room Shape

Adding Exits

Testing the Program

Training Mission #1
Exploring the Space Station in 3D
Training Mission #2
Making Your Own Maps
Are You Fit to Fly?
Mission Debrief

5

PREPARING THE SPACE STATION EQUIPMENT

Creating a Simple Planets Dictionary
 Understanding the Difference Between a List and a Dictionary
 Making an Astronomy Cheat Sheet Dictionary
 Error-Proofing the Dictionary
 Training Mission #1
 Putting Lists Inside Dictionaries
 Extracting Information from a List Inside a Dictionary
 Training Mission #2
Making the Space Station Objects Dictionary
 Adding the First Objects in Escape
 Viewing Objects with the Space Station Explorer
 Designing a Room
 Training Mission #3
 Adding the Rest of the Objects
 Training Mission #4
Are You Fit to Fly?
 Mission Debrief

6

INSTALLING THE SPACE STATION EQUIPMENT

Understanding the Dictionary for the Scenery Data
Adding the Scenery Data
Adding the Perimeter Fence for the Planet Surface
Loading the Scenery into Each Room
Updating the Explorer to Tour the Space Station

Training Mission #1
Are You Fit to Fly?
Mission Debrief

7

MOVING INTO THE SPACE STATION

Arriving on the Space Station

Disabling the Room Navigation Controls in the EXPLORER
Section

Adding New Variables

Teleporting onto the Space Station

Adding the Movement Code

Understanding the Movement Code

Training Mission #1

Moving Between Rooms

Are You Fit to Fly?

Mission Debrief

8

REPAIRING THE SPACE STATION

Sending Information to a Function

Creating a Function that Receives Information

How It Works

Training Mission #1

Adding Variables for Shadows, Wall Transparency, and Colors

Deleting the EXPLORER Section

Adding the DISPLAY Section

Adding the Functions for Drawing Objects

Drawing the Room

Understanding the New draw() Function

Positioning the Room on Your Screen

Making the Front Wall Fade In and Out

Displaying Hints, Tips, and Warnings

Showing the Room Name When You Enter the Room

Are You Fit to Fly?

Mission Debrief

9

UNPACKING YOUR PERSONAL ITEMS

Adding the Props Information

Adding Props to the Room Map

Finding an Object Number from the Room Map

Picking Up Objects

 Picking Up Props

 Adding the Keyboard Controls

Adding the Inventory Functionality

 Displaying the Inventory

 Adding the Tab Keyboard Control

 Testing the Inventory

Dropping Objects

Training Mission #1

Examining Objects

Training Mission #2

Are You Fit to Fly?

Mission Debrief

10

MAKE YOURSELF USEFUL

Adding the Keyboard Control for Using Objects

Adding Standard Messages for Using Objects

Adding the Game Progress Variables

Adding the Actions for Specific Objects

Combining Objects

Training Mission #1

Adding the Game Completion Sequence

Exploring the Objects

Are You Fit to Fly?

11

ACTIVATING SAFETY DOORS

Planning Where to Put Safety Doors

Positioning the Doors

Adding Access Controls

Making the Doors Open and Close

Adding the Door Animation

Training Mission #1

Shutting the Timed Door

Adding a Teleporter

Training Mission #2

Activating the Airlock Security Door

Removing Exits for Your Own Game Designs

Mission Accomplished?

Are You Fit to Fly?

12

DANGER! DANGER! ADDING HAZARDS

Adding the Air Countdown

Displaying the Air and Energy Bars

Adding the Air Countdown Functions

Starting the Air Countdown and Sounding the Alarm

Training Mission #1

Adding the Moving Hazards

Adding the Hazard Data

Sapping the Player's Energy

Starting and Stopping Hazards

Setting Up the Hazard Map

Making the Hazards Move

Displaying Hazards in the Room

Training Mission #2

Stopping the Player from Walking Through Hazards

Adding the Toxic Spills

Making the Finishing Touches

Disabling the Teleporter

Cleaning Up the Data
Your Adventure Begins
Your Next Mission: Customizing the Game
Are You Fit to Fly?
Mission Debrief

A
ESCAPE: THE COMPLETE GAME LISTING

B
TABLE OF VARIABLES, LISTS, AND DICTIONARIES

C
DEBUGGING YOUR LISTINGS

Indentation

Case Sensitivity

Parentheses and Brackets

Colons

Commas

Images and Sounds

Spelling

INDEX

ACKNOWLEDGMENTS

Many thanks to everyone at No Starch Press who worked hard to bring you this book, including developmental editor Liz Chadwick, production editor Riley Hoffman, copyeditor Anne Marie Walker, proofreaders Emelie Burnette and Meg Sneeringer, and production manager Serena Yang. Thank you to Tyler Ortman, who commissioned the book, and Bill Pollock, for his support on this project. Josh Ellingson created the stunning cover artwork. Thank you to Amanda Hariri, Anna Morrow, and Rachel Barry for their support with marketing.

Rafael Pimenta designed the awesome graphics for the game. Daniel Aldred was the technical editor, testing the code and providing feedback on the text. Thanks to them both.

We wouldn't have been able to create this book without the dedicated work of the open source community. Daniel Pope created Pygame Zero and helped with research queries. You can learn about some more cool features of Pygame Zero that weren't required for our mission at <http://pygame-zero.readthedocs.io/en/latest/>. Pygame Zero extends Pygame, so thanks also to the Pygame development team and to the wider Python community who contribute to its success.

NASA allows us to use many of its images to tell our story, for which we are grateful. Its work is hugely inspiring.

Thank you to Russell Barnes, Sam Alder, Eben Upton, and Carrie Anne Philbin at the Raspberry Pi Foundation who helped to get this project off the ground.

Finally, thank you for reading the book! If you enjoy it, please consider sharing a review, tweet, or blog post to help others to discover it. In any event, I hope you enjoy it.

INTRODUCTION



Air is running out. There's a leak in the space station, so you've got to act fast. Can you find your way to safety? You'll need to navigate your way around the space station, find access cards to unlock doors, and fix your damaged space suit. The adventure has begun!

And it starts here: on Earth, at mission command, also known as your computer. This book shows you how to use Python to build a space station on Mars, explore the station, and escape danger in an adventure game complete with graphics. Can you think like an astronaut to make it to safety?

HOW TO USE THIS BOOK

By following the instructions in this book, you can build a game called *Escape* with a map to explore and puzzles to solve. It's written in Python, a popular programming language that is easy to read. It also uses Pygame Zero, which adds some instructions for managing images and sounds, among other things. Bit by bit, I'll show you how to make the game and how the main parts of the code work, so you can customize it or build your own games based on my game code. You can also download all the code you need. If you get stuck or just want to jump straight into playing the game and seeing it work, you can do so. All the software you need is free, and I've provided instructions for Windows PCs and the Raspberry Pi. I recommend you use the Raspberry Pi 3 or Raspberry Pi 2. The game may run too slowly to enjoy on the Pi Zero, original Model B+, and older models.

There are several different ways you can use the book and the game:

- **Download the game, play it first, and then use the book to understand how it works.** This way, you eliminate the risk of seeing any spoilers in the book before you play the game! Although I've kept them to a minimum, you might notice a few clues in the code as you read the book. If you get really stuck on a problem in the game, you can try reading the code to work out the solution. In any case, I recommend you run the game at least once to see what you'll be building and learn how to run your programs.
- **Build the game, and then play it.** This book guides you through creating the game from start to finish. As you work your way through the chapters, you'll add new sections to the game and see how they work. If you can't get the code working at any point, you can just use my version of the code listing and continue building from there. If you choose this route, avoid making any custom changes to the game until you've built it, played it, and finished it. Otherwise, you might accidentally make the game impossible to complete. (It's okay to make any changes I suggest in the exercises.)
- **Customize the game.** When you understand how the program works, you can change it by using your own maps, graphics, objects, and puzzles. The *Escape* game is set on a space station, but yours could be in the jungle, under the sea, or almost anywhere. You could use the book to build your own version of *Escape* first, or use my version of the final game and customize that. I'd love to see what you make using the program as a starting point! You can find me on Twitter at @musicandwords or visit my website at www.sean.co.uk.

WHAT'S IN THIS BOOK?

Here's a briefing on what's in store for you as you embark on your mission.

- **Chapter 1** shows you how to go on a spacewalk. You'll learn how to use graphics in your Python programs using Pygame Zero and discover some of the basics of making Python programs.
- **Chapter 2** introduces *lists*, which store much of the information in the

Escape game. You'll see how to use lists to make a map.

- **Chapter 3** shows you how to get parts of a program to repeat and how to use that knowledge to display a map. You'll also design a room layout for the space station, using wall pillars and floor tiles.
- In **Chapter 4**, you'll start to build the *Escape* game, laying down the blueprints for the station. You'll see how the program understands the station layout and uses it to create the fabric for the rooms, putting the walls and floor in place.
- In **Chapter 5**, you'll learn how to use *dictionaries* in Python, which are another important way of storing information. You'll add information for all the objects the game uses, and you'll see how to create a preview of your own room design. When you extend the program in **Chapter 6**, you'll see all the scenery in place and will be able to look at all the rooms.
- After building the space station, you can move in. In **Chapter 7**, you'll add your astronaut character and discover how to move around the rooms and animate movements.
- **Chapter 8** shows you how to polish the game's graphics with shadows, fading walls, and a new function to draw the rooms that fixes the remaining graphical glitches.
- When the space station is operational, you can unpack your personal effects. In **Chapter 9**, you'll position items the player can examine, pick up, and drop. In **Chapter 10**, you'll see how to use and combine items, so you can solve puzzles in the game.
- The space station is nearly complete. **Chapter 11** adds safety doors that restrict access to certain zones. Just as you're putting your feet up and celebrating a job well done, there's danger around the corner, as you'll add moving hazards in **Chapter 12**.

As you work through the book, you'll complete training missions that give you an opportunity to test your programs and your coding skills. The answers, if you need them, are at the end of each chapter.

The appendixes at the back of the book will help you, too. **Appendix A** contains the listing for the whole game. If you're not sure where to add a new chunk of code, you can check here. **Appendix B** contains a table of the most important variables, lists, and dictionaries if you can't remember what's

stored where, and **Appendix C** has some debugging tips if a program doesn't work for you.

For more information and supporting resources for the book, visit the book's website at www.sean.co.uk/books/missionpython/. You can also find information and resources at <https://nostarch.com/missionpython/>.

INSTALLING THE SOFTWARE

The game uses the Python programming language and Pygame Zero, which is software that makes it easier to handle graphics and sound. You need to install both of these before you begin.

NOTE

For updated installation instructions, visit the book's web page at <https://nostarch.com/missionpython/>.

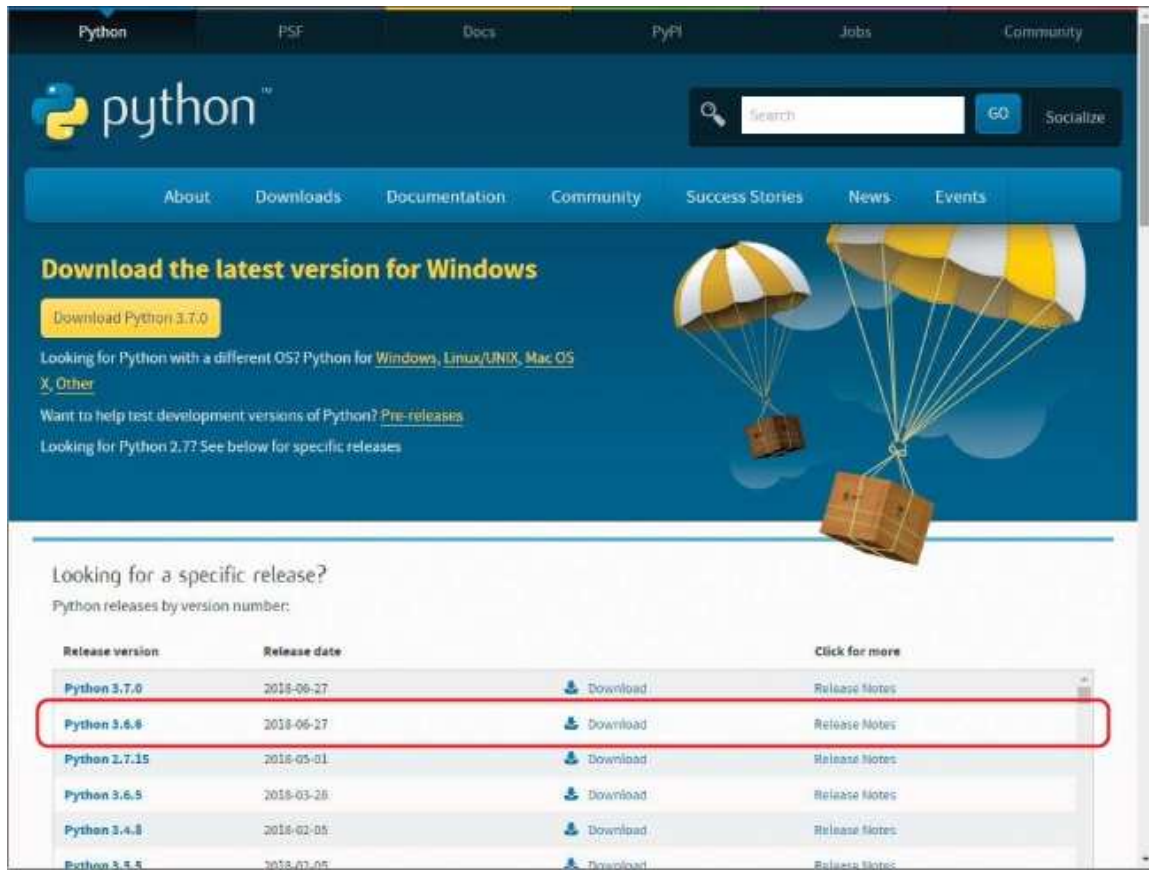
INSTALLING THE SOFTWARE ON RASPBERRY PI

If you're using a Raspberry Pi, Python and Pygame Zero are already installed. You can skip ahead to "Downloading the Game Files" on page 7.

INSTALLING PYTHON ON WINDOWS

To install the software on a Windows PC, follow these steps:

1. Open your web browser and visit <https://www.python.org/downloads/>.
2. At the time of this writing, 3.7 is the latest version of Python, but Pygame isn't available for easy installation on it yet. I recommend you use the latest version of Python 3.6 instead (3.6.6 at the time of writing). You can find old versions of Python farther down the screen on the downloads page (see Figure 1). Save the file on your desktop or somewhere else you can easily find it. (Pygame Zero works only with Python 3, so if you usually use Python 2, you'll need to switch to Python 3 for this book.)



Python PSF Docs PyPI Jobs Community

python™ Search GO Socialize

About Downloads Documentation Community Success Stories News Events

Download the latest version for Windows

[Download Python 3.7.0](#)

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)

Want to help test development versions of Python? [Pre-releases](#)

Looking for Python 2.7? See below for specific releases

Looking for a specific release?
Python releases by version number:

| Release version | Release date | | Click for more |
|-----------------|--------------|--------------------------|-------------------------------|
| Python 3.7.0 | 2018-06-27 | Download | Release Notes |
| Python 3.6.8 | 2018-06-27 | Download | Release Notes |
| Python 2.7.15 | 2018-05-01 | Download | Release Notes |
| Python 3.6.5 | 2018-03-28 | Download | Release Notes |
| Python 3.4.8 | 2018-02-05 | Download | Release Notes |
| Python 3.5.4 | 2018-01-05 | Download | Release Notes |

Figure 1: The Python downloads page

3. When the file has downloaded, double-click it to run it.
4. In the window that opens, select the checkbox to Add Python 3.6 to PATH (see Figure 2).
5. Click **Install Now**.



Figure 2: The Python installer

6. If you're asked whether you want to allow this application to make changes to your device, click **Yes**.
7. Python will take a few minutes to install. When it finishes, click **Close** to complete the installation.

INSTALLING PYGAME ZERO ON WINDOWS

Now that you have Python installed on your computer, you can install Pygame Zero. Follow these steps:

1. Hold down the **Windows Start key** and press **R**. The Run window should open (see Figure 3).
2. Enter `cmd` (see Figure 3). Press **ENTER** or click **OK**.

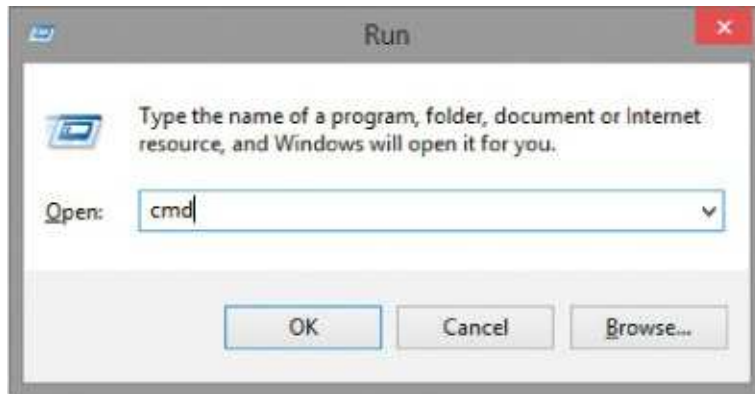


Figure 3: The Windows Run dialog box

3. The command line window should open, as shown in Figure 4. Here you can enter instructions for managing files or starting programs. Enter `pip install pygame` and press ENTER at the end of the line.

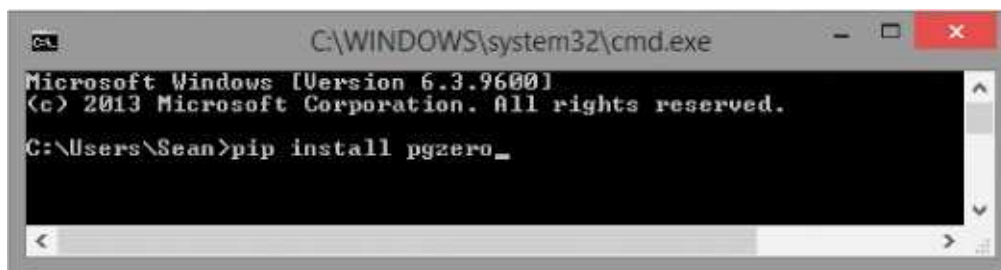


Figure 4: The command line window

4. Pygame Zero should start to install. It will take a few moments, and you'll know it's finished when your `>` prompt appears again.
5. If you get an error message saying that pip is not recognized, try installing Python again. You can uninstall Python first by running the installation program again or using the Windows Control panel. Make sure you select the box for the PATH when installing Python (see Figure 2). After you have reinstalled Python, try installing Pygame Zero again.
6. When Pygame Zero has finished downloading and you can type again, enter the following:

```
echo print("Hello!") > test.py
```

7. This line creates a new file called `test.py` that contains the instruction `print("Hello!")`. I'll explain the `print()` instruction in Chapter 1, but for now, this is just a quick way to make a test file. Be careful when you

enter the parentheses (curved brackets) and quotation marks: if you miss one, the file won't work properly.

8. Open the test file by entering the following:

```
pgzrun test.py
```

9. After a short delay, a blank window should open with the title *Pygame Zero Game*. Click the command line window again to bring it to the front: you should see the text `Hello!` Press `CTRL-C` in the command line window to stop the program.
10. If you want to delete your test program, enter `del test.py`.

INSTALLING THE SOFTWARE ON OTHER MACHINES

Python and Pygame Zero are available for other computer systems. Pygame Zero has been designed in part to enable games to work across different computers, so the *Escape* code should run wherever Pygame Zero runs. This book only provides guidance for users of Windows and Raspberry Pi computers. But if you have a different computer, you can download Python at <https://www.python.org/downloads/> and can find advice on installing Pygame Zero at <http://pygame-zero.readthedocs.io/en/latest/installation.html>.

DOWNLOADING THE GAME FILES

I've provided all the program files, sounds, and images you need for the *Escape* game. You can also download all the listings in the book, so if you can't get one to work, you can use mine instead. All the book's content downloads as a single ZIP file called *escape.zip*.

DOWNLOADING AND UNZIPPING THE FILES ON A RASPBERRY PI

To download the game files on a Raspberry Pi, follow these steps, and refer to Figure 5. The numbers in Figure 5 show you where to do each step.

- ❶ Open your web browser and visit <https://nostarch.com/missionpython/>. Click the link to download the files.

- ② From your desktop, click the File Manager icon on the taskbar at the top of the screen.
- ③ Double-click your Downloads folder to open it
- ④ Double-click the *escape.zip* file.
- ⑤ Click the **Extract Files** button to open the Extract Files dialog box.
- ⑥ Change the folder that you'll extract to so it reads *home/pi/escape*.
- ⑦ Ensure that the option is selected to Extract files with full path.
- ⑧ Click **Extract**.

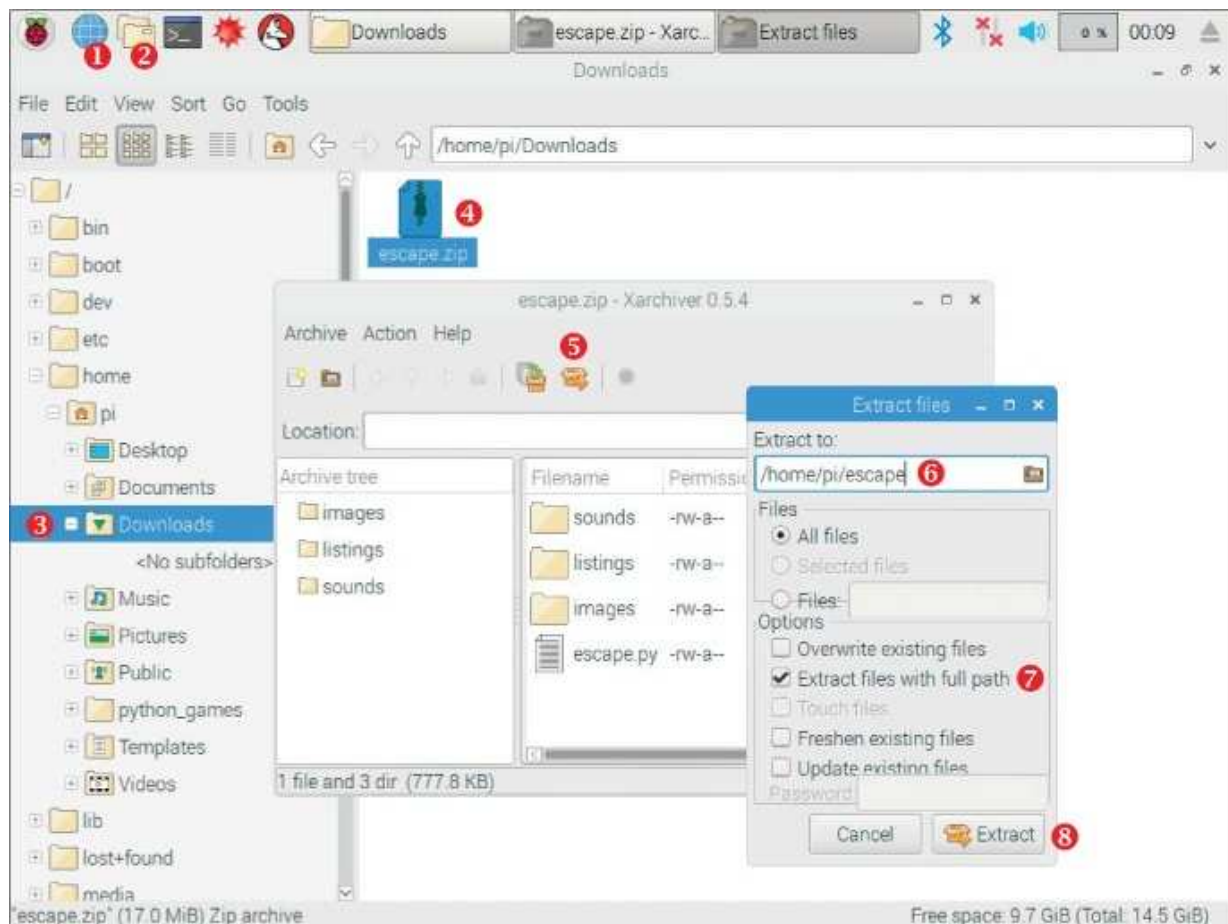


Figure 5: The steps you should take to unzip your files

UNZIPPING THE FILE ON A WINDOWS PC

To unzip the files on a Windows PC, follow these steps.

1. Open your web browser and visit <https://nostarch.com/missionpython/>. Click the link to download the files. Save the ZIP file on your desktop, in your *Documents* folder, or somewhere else you can easily find it.
2. Depending on the browser you're using, the ZIP file might open automatically, or there might be an option to open it at the bottom of the screen. If not, hold down the **Windows Start key** and press **E**. The Windows Explorer window should open. Go to the folder where you saved the ZIP file. Double-click the ZIP file.
3. Click **Extract All** at the top of the window.
4. I recommend that you create a folder called *escape* in your *Documents* folder and extract the files there. My documents folder is *C:\Users\Sean\Documents*, so I just typed *\escape* at the end of the folder name to create a new folder in that folder (see Figure 6). You can use the **Browse** button to get to your *Documents* folder first if necessary.
5. Click **Extract**.

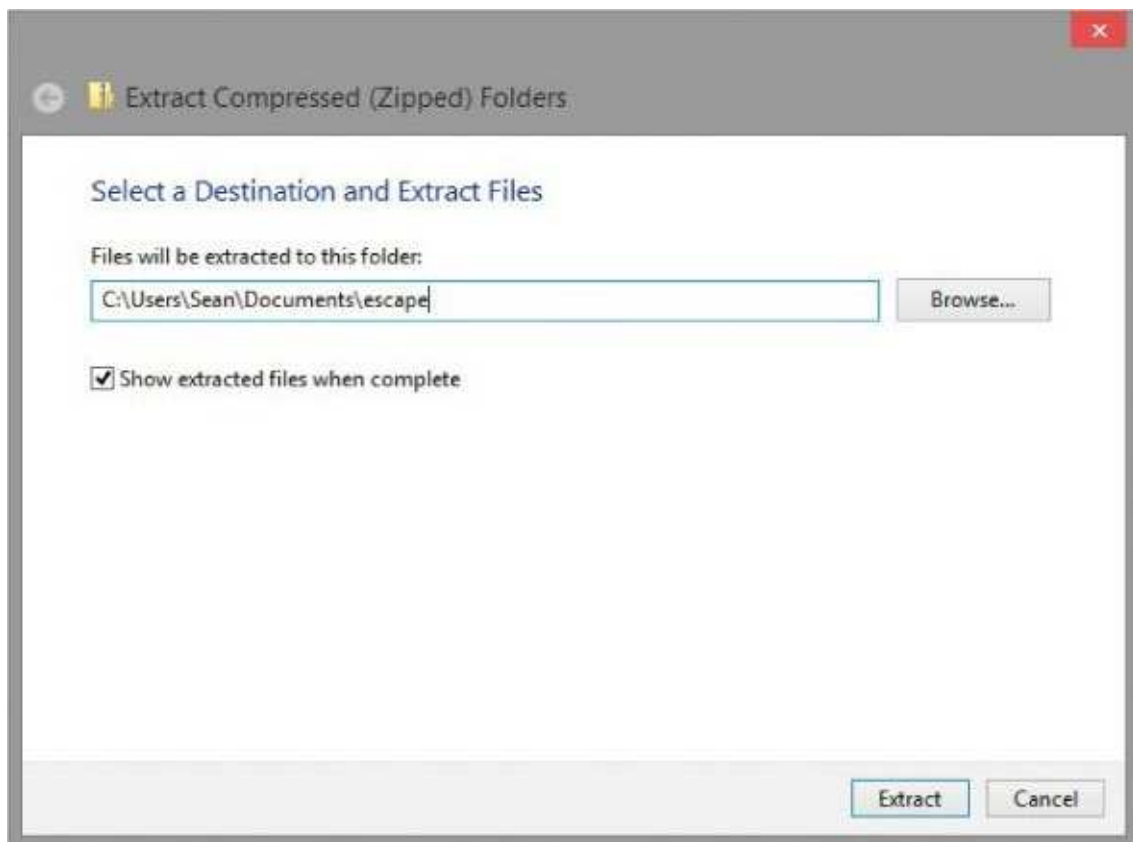


Figure 6: Setting the folder to unzip the game files into

WHAT'S IN THE ZIP FILE

The ZIP file you've just downloaded contains three folders and a Python program, *escape.py* (see Figure 7). The Python program is the final version of the *Escape* game, so you can start playing it right away. The *images* folder contains all the images you'll need for the game and other projects in this book. The *sounds* folder contains the sound effects.

In the *listings* folder, you'll find all the numbered listings in this book. If you can't get a program to work, try my version from this folder. You'll need to copy it from the listings folder first, and then paste it in the *escape* folder where the *escape.py* program is now. The reason you do this is because the program needs be alongside the *images* and *sounds* folders to work correctly.



Figure 7: The contents of the ZIP file as they might appear in Windows

RUNNING THE GAME

When you downloaded Python, another program called IDLE will have been downloaded with it. IDLE is an integrated development environment (IDE), which is software you can use to write programs in Python. You can run some of the listings in this book from the IDLE Python editor using the instructions provided. Most of the programs, though, use Pygame Zero, and you have to run those programs from the command line. Follow the instructions here to run the *Escape* game and any other Pygame Zero programs.

RUNNING PYGAME ZERO PROGRAMS ON THE RASPBERRY PI

If you're using a Raspberry Pi, follow these steps to run the *Escape* game:

1. Using the File Manager, go to your *escape* folder in your *pi* folder.
2. Click **Tools** on the menu and select **Open Current Folder in Terminal**, or you can press F4. The command line window (also known as the *shell*) should open, as shown in Figure 8. You can enter instructions here for managing files or starting programs.



Figure 8: The command line window on the Raspberry Pi

3. Type in the following command and press ENTER. The game begins!

```
pgzrun escape.py
```

This is how you run a Pygame Zero program on the Pi. To run the same program again, repeat the last step. To run a different program that's saved in the same folder, repeat the last step but change the name of the filename after `pgzrun`. To run a Pygame Zero program in a different folder, follow the steps starting from step 1, but open the command line from the folder with the program you want to run.

RUNNING PYGAME ZERO PROGRAMS IN WINDOWS

If you're using Windows, follow these steps to run the program:

1. Go to your *escape* folder. (Hold down the **Windows Start** key and press **E** to open the Windows Explorer again.)
2. Click the long bar above your files, as shown in Figure 9. Type `cmd` into this bar and press ENTER.

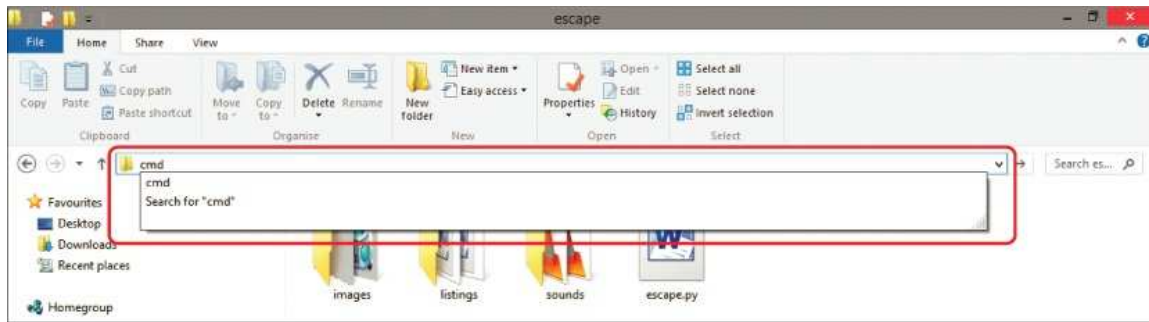


Figure 9: Finding the path to your Pygame files

3. The command line window will open. Your folder named *escape* will appear just before the `>` on the last line, as shown in Figure 10.

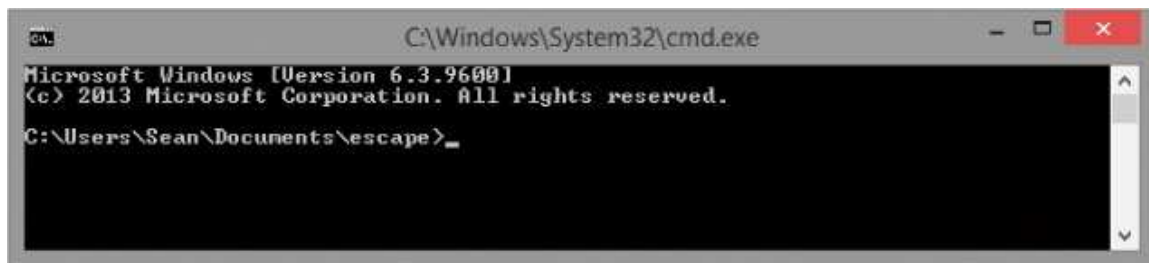


Figure 10: The command line window in Windows

4. Type `pgzrun escape.py` in the command line window. Press ENTER, and the *Escape* game begins.

This is how you run a Pygame Zero program on a Windows computer. You can run the program again by repeating the last step. To run a different program that's saved in the same folder, repeat the last step but change the name of the filename after `pgzrun`. To run a Pygame Zero program in a different folder, follow the steps starting from step 1, but open the command line from the folder with the program you want to run.

PLAYING THE GAME

You're working alone on the space station on Mars, many millions of kilometers from home. The rest of the crew is on a long-distance mission, exploring a canyon for signs of life, and won't be back for days. The murmuring hum of the life support systems surrounds you.

You're startled when the alarm sounds! There's a breach in the space station wall, and your air is slowly venting into the Martian atmosphere. You

climb quickly but carefully into your space suit, but the computer tells you the suit is damaged. Your life is at risk.

Your first priority is to repair your suit and ensure a reliable air supply. Your second priority is to radio for help, but the space station's radio systems are malfunctioning. Last night the Poodle lander, sent from Earth, crash-landed in the Martian dust. If you can find it, perhaps you can use its radio to issue a distress signal.

Use the arrow keys to move around the space station. To examine an object, stand on it and press the spacebar. Alternatively, if the object is something you can't walk on, press the spacebar while walking into it.

To pick up an object, walk onto it and press the G key (for *get*).

To select an object in your inventory, shown at the top of the screen (see Figure 11), press the TAB key to move through the items. To drop the selected object, press D.



Figure 11: Your adventure begins!

To use an object, either select it in your inventory or walk onto or into it and press U. You can combine objects or use them together when you press U while you carry one object and stand on the other or while you carry one and walk into the other.

You'll need to work out how to use your limited resources creatively to overcome obstacles and get to safety. Good luck!

1

YOUR FIRST SPACEWALK



Welcome to the space corps. Your mission is to build the first human outpost on Mars. For years, the world's greatest scientists have been sending robots to study it up close. Soon you too will set foot on its dusty surface.

Travel to Mars takes between six and eight months, depending on how Earth and Mars are aligned. During the journey, the spaceship risks hitting meteoroids and other space debris. If any damage occurs, you'll need to put on your spacesuit, go to the airlock, and then step into the void of space to make repairs, similar to the astronaut in Figure 1-1.

In this chapter, you'll go on a spacewalk by using Python to move a character around the screen. You'll launch your first Python program and learn some of the essential Python instructions you'll need to build the space station later in the book. You'll also learn how to create a sense of depth by overlapping images, which will prove essential when we create the *Escape* game in 3D later (starting with our first room mock-up in Chapter 3).



Figure 1-1: NASA astronaut Rick Mastracchio on a 26-minute spacewalk in 2010, as photographed by astronaut Clayton Anderson. The spacewalk outside the International Space Station was one of a series to replace coolant tanks.

If you haven't already installed Python and Pygame Zero (Windows users), see "Installing the Software" on page 3. You'll also need the *Escape* game files in this chapter. "Downloading the Game Files" on page 7 tells you how to download and unzip those files.

STARTING THE PYTHON EDITOR

As I mentioned in the Introduction, in this book we'll use the Python programming language. A programming language provides a way to write instructions for a computer. Our instructions will tell the computer how to do things like react to a keypress or display an image. We'll also be using Pygame Zero, which gives Python some additional instructions for handling sound and images.

Python comes with the IDLE editor, and we'll use the editor to create our Python programs. Because you've already installed Python, IDLE

should now be on your computer as well. The following sections explain how to start IDLE, depending on the type of computer you're using.

STARTING IDLE IN WINDOWS 10

To start IDLE in Windows 10, follow these steps:

1. Click the Cortana search box at the bottom of the screen, and enter **Python** in the box.
2. Click **IDLE** to open it.
3. With IDLE running, right-click its icon in the taskbar at the bottom of the screen and pin it. Then you can run it from there in the future using a single click.

STARTING IDLE IN WINDOWS 8

To start IDLE in Windows 8, follow these steps:

1. Move your mouse to the top right of the screen to show the Charms bar.
2. Click the Search icon, and enter **Python** in the box.
3. Click **IDLE** to open it.
4. With IDLE running, right-click its icon in the taskbar at the bottom of the screen and pin it. Then you can run it from there in the future using a single click.

STARTING IDLE ON THE RASPBERRY PI

To start IDLE on the Raspberry Pi, follow these steps:

1. Click the Programs menu at the top left of the screen.
2. Find the Programming category.
3. Click the Python 3 (IDLE) icon. The Raspberry Pi has both Python 2 and Python 3 installed, but most of the programs in this book will work only in Python 3.



- Lituz.com

Elektron kitoblar

**To'liq qismini Shu tugmani
bosish orqali sotib oling!**