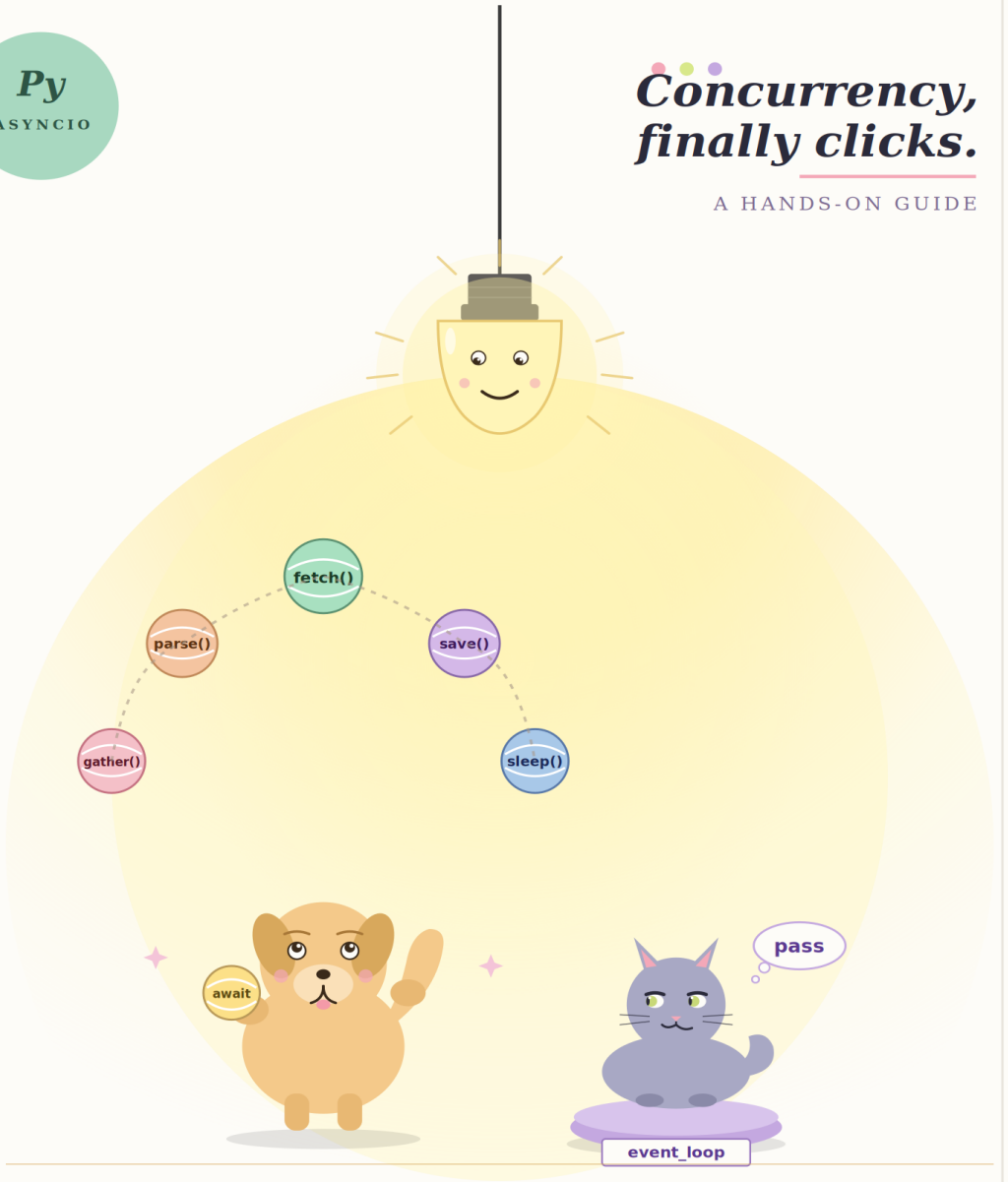


**Concurrency,  
finally clicks.**

A HANDS-ON GUIDE



# asyncio from groundup

*A working mental model for python asyncio*

FIRST  
EDITION  
2026

**RITESH MODI**

Head of AI, MarketOnce  
Ex-Microsoft Principal Engineer



# asyncio from ground up

A working Mental Model for Python asyncio

Ritesh Modi

This book is available at <https://leanpub.com/asyncio>

This version was published on 2026-05-04



This is a [Leanpub](#) book. Leanpub empowers authors and publishers with the Lean Publishing process. [Lean Publishing](#) is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

© 2026 Ritesh Modi

# Contents

<b>Getting Started</b> . . . . .	<b>1</b>
Unsure How to Get Started? Try our Book Workshop! . . . . .	1
How to Write on Leanpub . . . . .	1
Previewing and publishing . . . . .	1
Basic formatting . . . . .	1
Markdown and Markua . . . . .	1
Generate a preview version of your book . . . . .	1
Either read a tutorial, or just go for it! . . . . .	2
Thanks for being a Leanpub author! . . . . .	2
<b>Writing in Markua</b> . . . . .	<b>3</b>
Section One . . . . .	3
Including a Chapter in the Sample Book . . . . .	3
Links . . . . .	3
Images . . . . .	3
Lists . . . . .	8
Page Breaks . . . . .	9
Code Samples . . . . .	10
Tables . . . . .	11
Math . . . . .	11
Headings . . . . .	12
Block quotes, Asides and Blurbs . . . . .	13
Good luck, have fun! . . . . .	15
<b>From the Author</b> . . . . .	<b>17</b>
<b>Preface</b> . . . . .	<b>18</b>
<b>What This Book Is About</b> . . . . .	<b>19</b>
<b>What This Book Is Not</b> . . . . .	<b>20</b>

CONTENTS

<b>Who This Book Is For</b> . . . . .	<b>21</b>
<b>How to Use This Book</b> . . . . .	<b>22</b>
<b>Conventions Used in This Book</b> . . . . .	<b>23</b>
<b>Companion Code and Data</b> . . . . .	<b>24</b>
<b>How to Contact Us</b> . . . . .	<b>25</b>
<b>Acknowledgments</b> . . . . .	<b>26</b>
<b>Contents</b> . . . . .	<b>27</b>
<b>The Blocking Function</b> . . . . .	<b>28</b>
1.1 The first version of <code>news.py</code> . . . . .	28
1.2 What “blocking” actually means . . . . .	28
1.3 The cost grows with the list . . . . .	28
1.4 The same shape, one layer down . . . . .	28
1.5 The question that drives the rest of the book . . . . .	28
Summary . . . . .	28
<b>Concurrency, Parallelism, and the GIL</b> . . . . .	<b>30</b>
2.1 What a thread is, briefly . . . . .	30
2.2 The threaded version of <code>news.py</code> . . . . .	30
2.3 Threading the TCP probes . . . . .	30
2.4 Why the speedup actually works (and the GIL story you have heard wrong) . . . . .	30
2.5 Why threads are not the answer the book is reaching for . . . . .	30
2.6 The two scales the book cares about . . . . .	30
2.7 Where the next chapter goes . . . . .	31
Summary . . . . .	31
<b>The Loop, From Scratch</b> . . . . .	<b>32</b>
3.1 A queue of jobs . . . . .	32
3.2 A loop that runs the next ready job . . . . .	32
3.3 Two senses of “ready”, and the generator that gives us pause . . . . .	32
3.4 The loop sleeps when nothing is ready . . . . .	32
3.5 Connecting the toy loop to <code>news.py</code> . . . . .	32
Summary . . . . .	33
<b>Coroutines, Without the Magic</b> . . . . .	<b>34</b>

## CONTENTS

4.1 Generators, the original coroutine . . . . .	34
4.2 yield as a pause point, one more time . . . . .	34
4.3 The toy loop . . . . .	34
4.4 What async def returns . . . . .	34
Polite driver for <code>x in gen():</code> (no for; the loop handles it) . . . . .	34
. . . . .	34
4.5 Coroutines are not running until the loop runs them . . . . .	35
Summary . . . . .	35
<b>async and await . . . . .</b>	<b>36</b>
5.1 What async does to a function definition . . . . .	36
5.2 What await does at a call site . . . . .	36
5.3 What can be awaited, and what cannot . . . . .	36
5.4 The first real <code>asyncio.news.py</code> . . . . .	36
5.5 The cliffhanger . . . . .	36
5.6 The same keywords, on raw TCP . . . . .	37
Summary . . . . .	37
<b>await Yields, Suspends, and Resumes . . . . .</b>	<b>38</b>
6.1 What await does . . . . .	38
6.2 What is saved across the pause . . . . .	38
6.3 How the loop wakes the coroutine back up . . . . .	38
6.4 A Task is a coroutine that the loop is driving . . . . .	38
6.5 <i>For the curious:</i> <code>coro.send(value)</code> and the bytecode . . . . .	38
6.6 <i>For the curious:</i> selectors and the operating system . . . . .	38
6.7 Watching <code>news.py</code> yield and resume . . . . .	39
Summary . . . . .	39
<b>asyncio.run, create_task, gather . . . . .</b>	<b>40</b>
7.1 <code>asyncio.run</code> as the starting line . . . . .	40
7.2 What <code>asyncio.run</code> does . . . . .	40
7.3 <code>await coro</code> is sequential; <code>create_task(coro)</code> is concurrent . . . . .	40
7.4 <code>asyncio.gather</code> for “run these together” . . . . .	40
7.5 The fast <code>news.py</code> . . . . .	40
7.6 Concurrent TCP probes . . . . .	40
7.7 The lost-task trap . . . . .	41
Summary . . . . .	41
<b>The Time Toolbox . . . . .</b>	<b>42</b>
8.1 <code>asyncio.sleep</code> yields; <code>time.sleep</code> blocks . . . . .	42

## CONTENTS

8.2	<code>wait_for</code> for a deadline around an awaitable . . . . .	42
8.3	<code>asyncio.timeout()</code> as a context manager (Python 3.11+) . . . . .	42
8.4	<code>asyncio.shield</code> to protect from cancellation . . . . .	42
8.5	<code>asyncio.wait</code> vs <code>asyncio.gather</code> . . . . .	42
8.6	<code>asyncio.as_completed</code> for results as they arrive . . . . .	42
8.7	Timeouts and streaming . . . . .	43
	Summary . . . . .	43
	<b>async for, async with, Async Iteration and Resource Management . . .</b>	<b>44</b>
9.1	async for and the iterator protocol . . . . .	44
9.2	Async generators . . . . .	44
9.3	async for over <code>aiohttp</code> response bodies . . . . .	44
9.4	async with and the context-manager protocol . . . . .	44
9.5	Writing your own async context manager . . . . .	45
9.6	The streaming <code>news.py</code> . . . . .	45
	Summary . . . . .	45
	<b>Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars . .</b>	<b>46</b>
10.1	<code>Task</code> is the loop's unit of work; the coroutine is just the recipe . .	46
10.2	<code>asyncio.TaskGroup</code> : structured concurrency by construction . . .	46
10.3	Cancellation: <code>task.cancel()</code> and <code>CancelledError</code> at the next <code>await</code> . . . . .	46
10.4	The bare- <code>except</code> trap . . . . .	46
10.5	<code>ExceptionGroup</code> and <code>except*</code> . . . . .	46
10.6	Re-raising <code>CancelledError</code> after cleanup . . . . .	47
10.7	Canceling a TCP probe . . . . .	47
10.8	Synchronization primitives . . . . .	47
10.9	Capping concurrency with <code>Semaphore</code> : 50 sites, 10 in flight . . . .	48
10.10	<code>contextvars</code> for task-scoped state . . . . .	48
10.11	Producer-consumer with <code>asyncio.Queue</code> . . . . .	48
	Summary . . . . .	48
	<b>What Goes Wrong, and How to Find It . . . . .</b>	<b>49</b>
11.1	The failure mode: one synchronous call freezes the whole loop . .	49
11.2	The blocking patterns to learn to recognize . . . . .	49
11.3	The single most useful tool: <code>asyncio</code> 's debug mode . . . . .	49
11.4	The four warnings the runtime emits . . . . .	49
11.5	<code>asyncio.all_tasks()</code> and <code>task.get_stack()</code> . . . . .	49
11.6	<code>aiomonitor</code> : a live REPL inside a running program . . . . .	49

## CONTENTS

11.7 Profiling that sees the loop correctly . . . . .	50
11.8 Logging task names . . . . .	50
11.9 pdb and async . . . . .	50
11.10 A guided debugging session on the broken news.py . . . . .	50
11.11 tcpcheck.py as a diagnostic tool . . . . .	50
Summary . . . . .	50
<b>Testing async code . . . . .</b>	<b>51</b>
12.1 Your first async test . . . . .	51
12.2 The standard-library alternative . . . . .	51
12.3 What mocks are, and why async needs a different one . . . . .	51
12.4 Testing cancellation . . . . .	51
12.5 <i>Optional</i> : Testing TaskGroup error propagation . . . . .	51
12.6 Event loop fixture scope . . . . .	51
12.7 Testing timeouts without real time . . . . .	52
12.8 The most subtle async test bug . . . . .	52
12.9 When a test hangs . . . . .	52
12.10 The full test suite for the fetcher . . . . .	52
Summary . . . . .	52
<b>Where asyncio Ends: Executors, Graceful Shutdown . . . . .</b>	<b>53</b>
13.1 Fixing the parser bug from Chapter 11 . . . . .	53
13.2 The executor toolbox . . . . .	53
13.3 The decision tree . . . . .	53
Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload. . . . .	53
13.4 Bridging from sync to async, and back . . . . .	53
13.5 Library design: expose async, do not hide it . . . . .	54
13.6 Graceful shutdown . . . . .	54
13.7 The final news.py . . . . .	54
Summary . . . . .	54
<b>Appendix A: Setting up your environment . . . . .</b>	<b>56</b>
A.1 Python versions and what they buy you . . . . .	56
A.2 The packages the book uses . . . . .	56
A.3 The smallest runnable asyncio script . . . . .	56
A.4 Reading the running example as you go . . . . .	56
<b>Appendix B: The asyncio API surface . . . . .</b>	<b>57</b>
Functions . . . . .	57

CONTENTS

`asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8 . . . . . 57

Classes and protocols . . . . . 57

`ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except*` (3.11+). 10 . . . . . 57

Loop methods worth knowing . . . . . 57

`loop.set_default_executor(executor)` Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13 . . . . . 58

`aihttp` API surface used in the book . . . . . 58

`response.text()` Awaitable that returns the full response body as a decoded string. 5 . . . . . 58

Where to find the rest . . . . . 58

**Appendix C: Troubleshooting guide . . . . . 59**

C.1 Asyncio runtime warnings, what they mean and what to fix . . . . 59

`RuntimeWarning: async generator 'X' was never closed` An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between yields in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.) . . . . . 59

C.2 The script runs without errors but is no faster than synchronous . 59

Slow-callback warnings fire (`Executing <Task ...> took 0.234 seconds`) when running with `debug=True`. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.) . . . . . 59

C.3 The program hangs or never exits . . . . . 60

CONTENTS

Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.) . . . . . 60

C.4 Common errors and what they mean . . . . . 60

RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup. . . . . 61

When the table does not have your symptom . . . . . 61

**Appendix D: Further reading** . . . . . **62**

    D.1 The asyncio source code . . . . . 62

    D.2 The async library ecosystem . . . . . 62

    D.3 The structured concurrency literature . . . . . 62

    D.4 The PEPs that shaped asyncio . . . . . 62

    D.5 Things this book did not cover, that you may want next . . . . . 62

**Glossary** . . . . . **63**

**About the Author** . . . . . **64**

**From the Author** . . . . . **65**

**Preface** . . . . . **66**

**What This Book Is About** . . . . . **67**

**What This Book Is Not** . . . . . **68**

**Who This Book Is For** . . . . . **69**

**How to Use This Book** . . . . . **70**

**Conventions Used in This Book** . . . . . **71**

**Companion Code and Data** . . . . . **72**

CONTENTS

<b>How to Contact Us</b> . . . . .	<b>73</b>
<b>Acknowledgments</b> . . . . .	<b>74</b>
<b>Contents</b> . . . . .	<b>75</b>
<b>The Blocking Function</b> . . . . .	<b>76</b>
1.1 The first version of <code>news.py</code> . . . . .	76
1.2 What “blocking” actually means . . . . .	76
1.3 The cost grows with the list . . . . .	76
1.4 The same shape, one layer down . . . . .	76
1.5 The question that drives the rest of the book . . . . .	76
Summary . . . . .	76
<b>Concurrency, Parallelism, and the GIL</b> . . . . .	<b>78</b>
2.1 What a thread is, briefly . . . . .	78
2.2 The threaded version of <code>news.py</code> . . . . .	78
2.3 Threading the TCP probes . . . . .	78
2.4 Why the speedup actually works (and the GIL story you have heard wrong) . . . . .	78
2.5 Why threads are not the answer the book is reaching for . . . . .	78
2.6 The two scales the book cares about . . . . .	78
2.7 Where the next chapter goes . . . . .	79
Summary . . . . .	79
<b>The Loop, From Scratch</b> . . . . .	<b>80</b>
3.1 A queue of jobs . . . . .	80
3.2 A loop that runs the next ready job . . . . .	80
3.3 Two senses of “ready”, and the generator that gives us pause . . . . .	80
3.4 The loop sleeps when nothing is ready . . . . .	80
3.5 Connecting the toy loop to <code>news.py</code> . . . . .	80
Summary . . . . .	81
<b>Coroutines, Without the Magic</b> . . . . .	<b>82</b>
4.1 Generators, the original coroutine . . . . .	82
4.2 <code>yield</code> as a pause point, one more time . . . . .	82
4.3 The toy loop . . . . .	82
4.4 What <code>async def</code> returns . . . . .	82
Polite driver for <code>x in gen()</code> : (no <code>for</code> ; the loop handles it) . . . . .	82
. . . . .	82

## CONTENTS

4.5 Coroutines are not running until the loop runs them . . . . .	83
Summary . . . . .	83
<b>async and await . . . . .</b>	<b>84</b>
5.1 What async does to a function definition . . . . .	84
5.2 What await does at a call site . . . . .	84
5.3 What can be awaited, and what cannot . . . . .	84
5.4 The first real asyncio news.py . . . . .	84
5.5 The cliffhanger . . . . .	84
5.6 The same keywords, on raw TCP . . . . .	85
Summary . . . . .	85
<b>await Yields, Suspends, and Resumes . . . . .</b>	<b>86</b>
6.1 What await does . . . . .	86
6.2 What is saved across the pause . . . . .	86
6.3 How the loop wakes the coroutine back up . . . . .	86
6.4 A Task is a coroutine that the loop is driving . . . . .	86
6.5 <i>For the curious: coro.send(value)</i> and the bytecode . . . . .	86
6.6 <i>For the curious: selectors and the operating system</i> . . . . .	86
6.7 Watching news.py yield and resume . . . . .	87
Summary . . . . .	87
<b>asyncio.run, create_task, gather . . . . .</b>	<b>88</b>
7.1 asyncio.run as the starting line . . . . .	88
7.2 What asyncio.run does . . . . .	88
7.3 await coro is sequential; create_task(coro) is concurrent . . . . .	88
7.4 asyncio.gather for “run these together” . . . . .	88
7.5 The fast news.py . . . . .	88
7.6 Concurrent TCP probes . . . . .	88
7.7 The lost-task trap . . . . .	89
Summary . . . . .	89
<b>The Time Toolbox . . . . .</b>	<b>90</b>
8.1 asyncio.sleep yields; time.sleep blocks . . . . .	90
8.2 wait_for for a deadline around an awaitable . . . . .	90
8.3 asyncio.timeout() as a context manager (Python 3.11+) . . . . .	90
8.4 asyncio.shield to protect from cancellation . . . . .	90
8.5 asyncio.wait vs asyncio.gather . . . . .	90
8.6 asyncio.as_completed for results as they arrive . . . . .	90
8.7 Timeouts and streaming . . . . .	91

## CONTENTS

Summary . . . . .	91
<b>async for, async with, Async Iteration and Resource Management . . .</b>	<b>92</b>
9.1 async for and the iterator protocol . . . . .	92
9.2 Async generators . . . . .	92
9.3 async for over aiohttp response bodies . . . . .	92
9.4 async with and the context-manager protocol . . . . .	92
9.5 Writing your own async context manager . . . . .	93
9.6 The streaming news.py . . . . .	93
Summary . . . . .	93
<b>Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars . .</b>	<b>94</b>
10.1 Task is the loop's unit of work; the coroutine is just the recipe . .	94
10.2 asyncio.TaskGroup: structured concurrency by construction . . .	94
10.3 Cancellation: task.cancel() and CancelledError at the next await . . . . .	94
10.4 The bare-except trap . . . . .	94
10.5 ExceptionGroup and except* . . . . .	94
10.6 Re-raising CancelledError after cleanup . . . . .	95
10.7 Canceling a TCP probe . . . . .	95
10.8 Synchronization primitives . . . . .	95
10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight . . . .	96
10.10 contextvars for task-scoped state . . . . .	96
10.11 Producer-consumer with asyncio.Queue . . . . .	96
Summary . . . . .	96
<b>What Goes Wrong, and How to Find It . . . . .</b>	<b>97</b>
11.1 The failure mode: one synchronous call freezes the whole loop . .	97
11.2 The blocking patterns to learn to recognize . . . . .	97
11.3 The single most useful tool: asyncio's debug mode . . . . .	97
11.4 The four warnings the runtime emits . . . . .	97
11.5 asyncio.all_tasks() and task.get_stack() . . . . .	97
11.6 aiomonitor: a live REPL inside a running program . . . . .	97
11.7 Profiling that sees the loop correctly . . . . .	98
11.8 Logging task names . . . . .	98
11.9 pdb and async . . . . .	98
11.10 A guided debugging session on the broken news.py . . . . .	98
11.11 tcpcheck.py as a diagnostic tool . . . . .	98
Summary . . . . .	98

<b>Testing async code</b>	<b>99</b>
12.1 Your first async test	99
12.2 The standard-library alternative	99
12.3 What mocks are, and why async needs a different one	99
12.4 Testing cancellation	99
12.5 <i>Optional</i> : Testing TaskGroup error propagation	99
12.6 Event loop fixture scope	99
12.7 Testing timeouts without real time	100
12.8 The most subtle async test bug	100
12.9 When a test hangs	100
12.10 The full test suite for the fetcher	100
Summary	100
<b>Where asyncio Ends: Executors, Graceful Shutdown</b>	<b>101</b>
13.1 Fixing the parser bug from Chapter 11	101
13.2 The executor toolbox	101
13.3 The decision tree	101
Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.	101
13.4 Bridging from sync to async, and back	101
13.5 Library design: expose async, do not hide it	102
13.6 Graceful shutdown	102
13.7 The final news.py	102
Summary	102
<b>Appendix A: Setting up your environment</b>	<b>104</b>
A.1 Python versions and what they buy you	104
A.2 The packages the book uses	104
A.3 The smallest runnable asyncio script	104
A.4 Reading the running example as you go	104
<b>Appendix B: The asyncio API surface</b>	<b>105</b>
Functions	105
asyncio.wait_for(coro, timeout) Wraps a single awaitable with a deadline; cancels the inner work and raises TimeoutError on expiry. 8	105
Classes and protocols	105

CONTENTS

ExceptionGroup The exception type that holds multiple exceptions raised by sibling Tasks in a TaskGroup. Caught with `except*` (3.11+). 10 . . . . . 105

Loop methods worth knowing . . . . . 105

`loop.set_default_executor(executor)` Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13 . . . . . 106

aiohhttp API surface used in the book . . . . . 106

`response.text()` Awaitable that returns the full response body as a decoded string. 5 . . . . . 106

Where to find the rest . . . . . 106

**Appendix C: Troubleshooting guide . . . . . 107**

C.1 Asyncio runtime warnings, what they mean and what to fix . . . . . 107

RuntimeWarning: async generator 'X' was never closed An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between yields in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.) . . . . . 107

C.2 The script runs without errors but is no faster than synchronous . 107

Slow-callback warnings fire (`Executing <Task ...> took 0.234 seconds`) when running with `debug=True`. A coroutine ran for too long between `awaits`, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.) . . . . . 107

C.3 The program hangs or never exits . . . . . 108

Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.) . . . . . 108

C.4 Common errors and what they mean . . . . . 108

## CONTENTS

RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after <code>asyncio.run</code> returned and closed it. Common when a finalizer or <code>__del__</code> method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (finally blocks, <code>__aexit__</code> ). Do not rely on <code>__del__</code> or <code>atexit</code> for async cleanup. . . . .	109
When the table does not have your symptom . . . . .	109
<b>Appendix D: Further reading</b> . . . . .	<b>110</b>
D.1 The asyncio source code . . . . .	110
D.2 The async library ecosystem . . . . .	110
D.3 The structured concurrency literature . . . . .	110
D.4 The PEPs that shaped asyncio . . . . .	110
D.5 Things this book did not cover, that you may want next . . . . .	110
<b>Glossary</b> . . . . .	<b>111</b>
<b>About the Author</b> . . . . .	<b>112</b>
<b>From the Author</b> . . . . .	<b>113</b>
<b>Preface</b> . . . . .	<b>114</b>
<b>What This Book Is About</b> . . . . .	<b>115</b>
<b>What This Book Is Not</b> . . . . .	<b>116</b>
<b>Who This Book Is For</b> . . . . .	<b>117</b>
<b>How to Use This Book</b> . . . . .	<b>118</b>
<b>Conventions Used in This Book</b> . . . . .	<b>119</b>
<b>Companion Code and Data</b> . . . . .	<b>120</b>
<b>How to Contact Us</b> . . . . .	<b>121</b>
<b>Acknowledgments</b> . . . . .	<b>122</b>
<b>Contents</b> . . . . .	<b>123</b>
<b>The Blocking Function</b> . . . . .	<b>124</b>

## CONTENTS

1.1 The first version of <code>news.py</code> . . . . .	124
1.2 What “blocking” actually means . . . . .	124
1.3 The cost grows with the list . . . . .	124
1.4 The same shape, one layer down . . . . .	124
1.5 The question that drives the rest of the book . . . . .	124
Summary . . . . .	124
<b>Concurrency, Parallelism, and the GIL . . . . .</b>	<b>126</b>
2.1 What a thread is, briefly . . . . .	126
2.2 The threaded version of <code>news.py</code> . . . . .	126
2.3 Threading the TCP probes . . . . .	126
2.4 Why the speedup actually works (and the GIL story you have heard wrong) . . . . .	126
2.5 Why threads are not the answer the book is reaching for . . . . .	126
2.6 The two scales the book cares about . . . . .	126
2.7 Where the next chapter goes . . . . .	127
Summary . . . . .	127
<b>The Loop, From Scratch . . . . .</b>	<b>128</b>
3.1 A queue of jobs . . . . .	128
3.2 A loop that runs the next ready job . . . . .	128
3.3 Two senses of “ready”, and the generator that gives us pause . . . . .	128
3.4 The loop sleeps when nothing is ready . . . . .	128
3.5 Connecting the toy loop to <code>news.py</code> . . . . .	128
Summary . . . . .	129
<b>Coroutines, Without the Magic . . . . .</b>	<b>130</b>
4.1 Generators, the original coroutine . . . . .	130
4.2 <code>yield</code> as a pause point, one more time . . . . .	130
4.3 The toy loop . . . . .	130
4.4 What <code>async def</code> returns . . . . .	130
Polite driver for <code>x in gen():</code> (no <code>for</code> ; the loop handles it) . . . . .	130
. . . . .	130
4.5 Coroutines are not running until the loop runs them . . . . .	131
Summary . . . . .	131
<b><code>async</code> and <code>await</code> . . . . .</b>	<b>132</b>
5.1 What <code>async</code> does to a function definition . . . . .	132
5.2 What <code>await</code> does at a call site . . . . .	132
5.3 What can be awaited, and what cannot . . . . .	132

## CONTENTS

5.4 The first real <code>asyncio</code> news.py . . . . .	132
5.5 The cliffhanger . . . . .	132
5.6 The same keywords, on raw TCP . . . . .	133
Summary . . . . .	133
<b>await Yields, Suspends, and Resumes . . . . .</b>	<b>134</b>
6.1 What <code>await</code> does . . . . .	134
6.2 What is saved across the pause . . . . .	134
6.3 How the loop wakes the coroutine back up . . . . .	134
6.4 A <code>Task</code> is a coroutine that the loop is driving . . . . .	134
6.5 <i>For the curious</i> : <code>coro.send(value)</code> and the bytecode . . . . .	134
6.6 <i>For the curious</i> : selectors and the operating system . . . . .	134
6.7 Watching news.py yield and resume . . . . .	135
Summary . . . . .	135
<b>asyncio.run, create_task, gather . . . . .</b>	<b>136</b>
7.1 <code>asyncio.run</code> as the starting line . . . . .	136
7.2 What <code>asyncio.run</code> does . . . . .	136
7.3 <code>await coro</code> is sequential; <code>create_task(coro)</code> is concurrent . . . . .	136
7.4 <code>asyncio.gather</code> for “run these together” . . . . .	136
7.5 The fast news.py . . . . .	136
7.6 Concurrent TCP probes . . . . .	136
7.7 The lost-task trap . . . . .	137
Summary . . . . .	137
<b>The Time Toolbox . . . . .</b>	<b>138</b>
8.1 <code>asyncio.sleep</code> yields; <code>time.sleep</code> blocks . . . . .	138
8.2 <code>wait_for</code> for a deadline around an awaitable . . . . .	138
8.3 <code>asyncio.timeout()</code> as a context manager (Python 3.11+) . . . . .	138
8.4 <code>asyncio.shield</code> to protect from cancellation . . . . .	138
8.5 <code>asyncio.wait</code> vs <code>asyncio.gather</code> . . . . .	138
8.6 <code>asyncio.as_completed</code> for results as they arrive . . . . .	138
8.7 Timeouts and streaming . . . . .	139
Summary . . . . .	139
<b>async for, async with, Async Iteration and Resource Management . . . . .</b>	<b>140</b>
9.1 <code>async for</code> and the iterator protocol . . . . .	140
9.2 Async generators . . . . .	140
9.3 <code>async for</code> over aiohttp response bodies . . . . .	140
9.4 <code>async with</code> and the context-manager protocol . . . . .	140

CONTENTS

- 9.5 Writing your own async context manager . . . . . 141
- 9.6 The streaming news.py . . . . . 141
- Summary . . . . . 141
- Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars . . 142**
  - 10.1 Task is the loop's unit of work; the coroutine is just the recipe . . 142
  - 10.2 asyncio.TaskGroup: structured concurrency by construction . . . 142
  - 10.3 Cancellation: task.cancel() and CancelledError at the next  
await . . . . . 142
  - 10.4 The bare-except trap . . . . . 142
  - 10.5 ExceptionGroup and except\* . . . . . 142
  - 10.6 Re-raising CancelledError after cleanup . . . . . 143
  - 10.7 Canceling a TCP probe . . . . . 143
  - 10.8 Synchronization primitives . . . . . 143
  - 10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight . . . . 144
  - 10.10 contextvars for task-scoped state . . . . . 144
  - 10.11 Producer-consumer with asyncio.Queue . . . . . 144
  - Summary . . . . . 144
- What Goes Wrong, and How to Find It . . . . . 145**
  - 11.1 The failure mode: one synchronous call freezes the whole loop . . 145
  - 11.2 The blocking patterns to learn to recognize . . . . . 145
  - 11.3 The single most useful tool: asyncio's debug mode . . . . . 145
  - 11.4 The four warnings the runtime emits . . . . . 145
  - 11.5 asyncio.all\_tasks() and task.get\_stack() . . . . . 145
  - 11.6 aiomonitor: a live REPL inside a running program . . . . . 145
  - 11.7 Profiling that sees the loop correctly . . . . . 146
  - 11.8 Logging task names . . . . . 146
  - 11.9 pdb and async . . . . . 146
  - 11.10 A guided debugging session on the broken news.py . . . . . 146
  - 11.11 tcpcheck.py as a diagnostic tool . . . . . 146
  - Summary . . . . . 146
- Testing async code . . . . . 147**
  - 12.1 Your first async test . . . . . 147
  - 12.2 The standard-library alternative . . . . . 147
  - 12.3 What mocks are, and why async needs a different one . . . . . 147
  - 12.4 Testing cancellation . . . . . 147
  - 12.5 *Optional*: Testing TaskGroup error propagation . . . . . 147

CONTENTS

- 12.6 Event loop fixture scope . . . . . 147
- 12.7 Testing timeouts without real time . . . . . 148
- 12.8 The most subtle async test bug . . . . . 148
- 12.9 When a test hangs . . . . . 148
- 12.10 The full test suite for the fetcher . . . . . 148
- Summary . . . . . 148
  
- Where asyncio Ends: Executors, Graceful Shutdown . . . . . 149**
- 13.1 Fixing the parser bug from Chapter 11 . . . . . 149
- 13.2 The executor toolbox . . . . . 149
- 13.3 The decision tree . . . . . 149
- Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload. . . . . 149
- 13.4 Bridging from sync to async, and back . . . . . 149
- 13.5 Library design: expose async, do not hide it . . . . . 150
- 13.6 Graceful shutdown . . . . . 150
- 13.7 The final news.py . . . . . 150
- Summary . . . . . 150
  
- Appendix A: Setting up your environment . . . . . 152**
- A.1 Python versions and what they buy you . . . . . 152
- A.2 The packages the book uses . . . . . 152
- A.3 The smallest runnable asyncio script . . . . . 152
- A.4 Reading the running example as you go . . . . . 152
  
- Appendix B: The asyncio API surface . . . . . 153**
- Functions . . . . . 153
- `asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8 . . . . . 153
- Classes and protocols . . . . . 153
- `ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except* (3.11+)`. 10 . . . . . 153
- Loop methods worth knowing . . . . . 153
- `loop.set_default_executor(executor)` Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13 . . . . . 154
- `aihttp` API surface used in the book . . . . . 154

response.text() Awaitable that returns the full response body as a decoded string. 5 . . . . .	154
Where to find the rest . . . . .	154
<b>Appendix C: Troubleshooting guide . . . . .</b>	<b>155</b>
C.1 Asyncio runtime warnings, what they mean and what to fix . . . . .	155
RuntimeWarning: async generator 'X' was never closed An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling aclose(). Wrap resources held between yields in try/finally. Call aclose() explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.) . . . . .	155
C.2 The script runs without errors but is no faster than synchronous .	155
Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with asyncio.to_thread. For CPU-bound work that needs cores, use a process pool via loop.run_in_executor(ProcessPoolExecutor(), ...). (Chapter 11; Chapter 13.) . . . . .	155
C.3 The program hangs or never exits . . . . .	156
Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but asyncio.run is blocked elsewhere. Install signal handlers with loop.add_ signal_handler(sig, callback). Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.) . . . . .	156
C.4 Common errors and what they mean . . . . .	156
RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after asyncio.run returned and closed it. Common when a finalizer or __del__ method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (finally blocks, __aexit__). Do not rely on __del__ or atexit for async cleanup. . . . .	157
When the table does not have your symptom . . . . .	157
<b>Appendix D: Further reading . . . . .</b>	<b>158</b>

## CONTENTS

D.1 The asyncio source code . . . . .	158
D.2 The async library ecosystem . . . . .	158
D.3 The structured concurrency literature . . . . .	158
D.4 The PEPs that shaped asyncio . . . . .	158
D.5 Things this book did not cover, that you may want next . . . . .	158
<b>Glossary . . . . .</b>	<b>159</b>
<b>About the Author . . . . .</b>	<b>160</b>
<b>From the Author . . . . .</b>	<b>161</b>
<b>Preface . . . . .</b>	<b>162</b>
<b>What This Book Is About . . . . .</b>	<b>163</b>
<b>What This Book Is Not . . . . .</b>	<b>164</b>
<b>Who This Book Is For . . . . .</b>	<b>165</b>
<b>How to Use This Book . . . . .</b>	<b>166</b>
<b>Conventions Used in This Book . . . . .</b>	<b>167</b>
<b>Companion Code and Data . . . . .</b>	<b>168</b>
<b>How to Contact Us . . . . .</b>	<b>169</b>
<b>Acknowledgments . . . . .</b>	<b>170</b>
<b>Contents . . . . .</b>	<b>171</b>
<b>The Blocking Function . . . . .</b>	<b>172</b>
1.1 The first version of news.py . . . . .	172
1.2 What “blocking” actually means . . . . .	172
1.3 The cost grows with the list . . . . .	172
1.4 The same shape, one layer down . . . . .	172
1.5 The question that drives the rest of the book . . . . .	172
Summary . . . . .	172
<b>Concurrency, Parallelism, and the GIL . . . . .</b>	<b>174</b>
2.1 What a thread is, briefly . . . . .	174
2.2 The threaded version of news.py . . . . .	174

CONTENTS

2.3 Threading the TCP probes . . . . .	174
2.4 Why the speedup actually works (and the GIL story you have heard wrong) . . . . .	174
2.5 Why threads are not the answer the book is reaching for . . . . .	174
2.6 The two scales the book cares about . . . . .	174
2.7 Where the next chapter goes . . . . .	175
Summary . . . . .	175
<b>The Loop, From Scratch . . . . .</b>	<b>176</b>
3.1 A queue of jobs . . . . .	176
3.2 A loop that runs the next ready job . . . . .	176
3.3 Two senses of “ready”, and the generator that gives us pause . . . . .	176
3.4 The loop sleeps when nothing is ready . . . . .	176
3.5 Connecting the toy loop to news.py . . . . .	176
Summary . . . . .	177
<b>Coroutines, Without the Magic . . . . .</b>	<b>178</b>
4.1 Generators, the original coroutine . . . . .	178
4.2 yield as a pause point, one more time . . . . .	178
4.3 The toy loop . . . . .	178
4.4 What async def returns . . . . .	178
Polite driver for <code>x in gen():</code> (no for; the loop handles it) . . . . .	178
. . . . .	178
4.5 Coroutines are not running until the loop runs them . . . . .	179
Summary . . . . .	179
<b>async and await . . . . .</b>	<b>180</b>
5.1 What async does to a function definition . . . . .	180
5.2 What await does at a call site . . . . .	180
5.3 What can be awaited, and what cannot . . . . .	180
5.4 The first real asyncio news.py . . . . .	180
5.5 The cliffhanger . . . . .	180
5.6 The same keywords, on raw TCP . . . . .	181
Summary . . . . .	181
<b>await Yields, Suspends, and Resumes . . . . .</b>	<b>182</b>
6.1 What await does . . . . .	182
6.2 What is saved across the pause . . . . .	182
6.3 How the loop wakes the coroutine back up . . . . .	182
6.4 A Task is a coroutine that the loop is driving . . . . .	182

## CONTENTS

6.5 <i>For the curious: coro.send(value) and the bytecode</i> . . . . .	182
6.6 <i>For the curious: selectors and the operating system</i> . . . . .	182
6.7 Watching news.py yield and resume . . . . .	183
Summary . . . . .	183
<b>asyncio.run, create_task, gather</b> . . . . .	<b>184</b>
7.1 asyncio.run as the starting line . . . . .	184
7.2 What asyncio.run does . . . . .	184
7.3 await coro is sequential; create_task(coro) is concurrent . . . . .	184
7.4 asyncio.gather for “run these together” . . . . .	184
7.5 The fast news.py . . . . .	184
7.6 Concurrent TCP probes . . . . .	184
7.7 The lost-task trap . . . . .	185
Summary . . . . .	185
<b>The Time Toolbox</b> . . . . .	<b>186</b>
8.1 asyncio.sleep yields; time.sleep blocks . . . . .	186
8.2 wait_for for a deadline around an awaitable . . . . .	186
8.3 asyncio.timeout() as a context manager (Python 3.11+) . . . . .	186
8.4 asyncio.shield to protect from cancellation . . . . .	186
8.5 asyncio.wait vs asyncio.gather . . . . .	186
8.6 asyncio.as_completed for results as they arrive . . . . .	186
8.7 Timeouts and streaming . . . . .	187
Summary . . . . .	187
<b>async for, async with, Async Iteration and Resource Management</b> . . . . .	<b>188</b>
9.1 async for and the iterator protocol . . . . .	188
9.2 Async generators . . . . .	188
9.3 async for over aiohttp response bodies . . . . .	188
9.4 async with and the context-manager protocol . . . . .	188
9.5 Writing your own async context manager . . . . .	189
9.6 The streaming news.py . . . . .	189
Summary . . . . .	189
<b>Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars</b> . . . . .	<b>190</b>
10.1 Task is the loop’s unit of work; the coroutine is just the recipe . . . . .	190
10.2 asyncio.TaskGroup: structured concurrency by construction . . . . .	190
10.3 Cancellation: task.cancel() and CancelledError at the next await . . . . .	190
10.4 The bare-except trap . . . . .	190

CONTENTS

- 10.5 ExceptionGroup and except\* . . . . . 190
- 10.6 Re-raising CancelledError after cleanup . . . . . 191
- 10.7 Canceling a TCP probe . . . . . 191
- 10.8 Synchronization primitives . . . . . 191
- 10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight . . . . 192
- 10.10 contextvars for task-scoped state . . . . . 192
- 10.11 Producer-consumer with asyncio.Queue . . . . . 192
- Summary . . . . . 192
  
- What Goes Wrong, and How to Find It . . . . . 193**
- 11.1 The failure mode: one synchronous call freezes the whole loop . . 193
- 11.2 The blocking patterns to learn to recognize . . . . . 193
- 11.3 The single most useful tool: asyncio's debug mode . . . . . 193
- 11.4 The four warnings the runtime emits . . . . . 193
- 11.5 asyncio.all\_tasks() and task.get\_stack() . . . . . 193
- 11.6 aiomonitor: a live REPL inside a running program . . . . . 193
- 11.7 Profiling that sees the loop correctly . . . . . 194
- 11.8 Logging task names . . . . . 194
- 11.9 pdb and asyncio . . . . . 194
- 11.10 A guided debugging session on the broken news.py . . . . . 194
- 11.11 tcpcheck.py as a diagnostic tool . . . . . 194
- Summary . . . . . 194
  
- Testing async code . . . . . 195**
- 12.1 Your first async test . . . . . 195
- 12.2 The standard-library alternative . . . . . 195
- 12.3 What mocks are, and why async needs a different one . . . . . 195
- 12.4 Testing cancellation . . . . . 195
- 12.5 *Optional*: Testing TaskGroup error propagation . . . . . 195
- 12.6 Event loop fixture scope . . . . . 195
- 12.7 Testing timeouts without real time . . . . . 196
- 12.8 The most subtle async test bug . . . . . 196
- 12.9 When a test hangs . . . . . 196
- 12.10 The full test suite for the fetcher . . . . . 196
- Summary . . . . . 196
  
- Where asyncio Ends: Executors, Graceful Shutdown . . . . . 197**
- 13.1 Fixing the parser bug from Chapter 11 . . . . . 197
- 13.2 The executor toolbox . . . . . 197

CONTENTS

- 13.3 The decision tree . . . . . 197
- Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload. . . . . 197
- 13.4 Bridging from sync to async, and back . . . . . 197
- 13.5 Library design: expose async, do not hide it . . . . . 198
- 13.6 Graceful shutdown . . . . . 198
- 13.7 The final news.py . . . . . 198
- Summary . . . . . 198
  
- Appendix A: Setting up your environment . . . . . 200**
- A.1 Python versions and what they buy you . . . . . 200
- A.2 The packages the book uses . . . . . 200
- A.3 The smallest runnable asyncio script . . . . . 200
- A.4 Reading the running example as you go . . . . . 200
  
- Appendix B: The asyncio API surface . . . . . 201**
- Functions . . . . . 201
- asyncio.wait\_for(coro, timeout) Wraps a single awaitable with a deadline; cancels the inner work and raises TimeoutError on expiry. 8 . . . . . 201
- Classes and protocols . . . . . 201
- ExceptionGroup The exception type that holds multiple exceptions raised by sibling Tasks in a TaskGroup. Caught with except\* (3.11+). 10 . . . . . 201
- Loop methods worth knowing . . . . . 201
- loop.set\_default\_executor(executor) Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13 . . . . . 202
- aihttp API surface used in the book . . . . . 202
- response.text() Awaitable that returns the full response body as a decoded string. 5 . . . . . 202
- Where to find the rest . . . . . 202
  
- Appendix C: Troubleshooting guide . . . . . 203**
- C.1 Asyncio runtime warnings, what they mean and what to fix . . . . . 203

CONTENTS

RuntimeWarning: async generator 'X' was never closed An  
async generator was garbage-collected mid-iteration without the  
consumer running it to completion or calling `aclose()`. Wrap  
resources held between yields in `try/finally`. Call `aclose()`  
explicitly when you need cleanup at a specific moment. (Chapter  
9, Section 9.2.1) . . . . . 203

C.2 The script runs without errors but is no faster than synchronous . 203

Slow-callback warnings fire (Executing `<Task ...>` took 0.234  
seconds) when running with `debug=True`. A coroutine ran  
for too long between `awaits`, blocking the loop. Usually  
a CPU-bound function or a sync I/O call. Move the slow  
work to a thread pool with `asyncio.to_thread`. For  
CPU-bound work that needs cores, use a process pool via  
`loop.run_in_executor(ProcessPoolExecutor(), ...)`.  
(Chapter 11; Chapter 13.) . . . . . 203

C.3 The program hangs or never exits . . . . . 204

Ctrl+C does not interrupt the program. No signal handler installed;  
or a signal handler that schedules async work but `asyncio.run`  
is blocked elsewhere. Install signal handlers with `loop.add_  
signal_handler(sig, callback)`. Cancel the top-level Task in  
the callback. The TaskGroup propagates cancellation to children.  
(Chapter 13.) . . . . . 204

C.4 Common errors and what they mean . . . . . 204

RuntimeError: Event loop is closed (often during program  
shutdown) Code is trying to use the loop after `asyncio.run`  
returned and closed it. Common when a finalizer or  
`__del__` method tries to schedule async work. Move any  
async cleanup into the running coroutine's normal exit path  
(`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit`  
for async cleanup. . . . . 205

When the table does not have your symptom . . . . . 205

**Appendix D: Further reading . . . . . 206**

D.1 The asyncio source code . . . . . 206

D.2 The async library ecosystem . . . . . 206

D.3 The structured concurrency literature . . . . . 206

D.4 The PEPs that shaped asyncio . . . . . 206

D.5 Things this book did not cover, that you may want next . . . . . 206

CONTENTS

<b>Glossary</b> . . . . .	207
<b>About the Author</b> . . . . .	208
<b>From the Author</b> . . . . .	209
<b>Preface</b> . . . . .	210
<b>What This Book Is About</b> . . . . .	211
<b>What This Book Is Not</b> . . . . .	212
<b>Who This Book Is For</b> . . . . .	213
<b>How to Use This Book</b> . . . . .	214
<b>Conventions Used in This Book</b> . . . . .	215
<b>Companion Code and Data</b> . . . . .	216
<b>How to Contact Us</b> . . . . .	217
<b>Acknowledgments</b> . . . . .	218
<b>Contents</b> . . . . .	219
<b>The Blocking Function</b> . . . . .	220
1.1 The first version of <code>news.py</code> . . . . .	220
1.2 What “blocking” actually means . . . . .	220
1.3 The cost grows with the list . . . . .	220
1.4 The same shape, one layer down . . . . .	220
1.5 The question that drives the rest of the book . . . . .	220
Summary . . . . .	220
<b>Concurrency, Parallelism, and the GIL</b> . . . . .	222
2.1 What a thread is, briefly . . . . .	222
2.2 The threaded version of <code>news.py</code> . . . . .	222
2.3 Threading the TCP probes . . . . .	222
2.4 Why the speedup actually works (and the GIL story you have heard wrong) . . . . .	222
2.5 Why threads are not the answer the book is reaching for . . . . .	222
2.6 The two scales the book cares about . . . . .	222
2.7 Where the next chapter goes . . . . .	223

CONTENTS

Summary . . . . .	223
<b>The Loop, From Scratch . . . . .</b>	<b>224</b>
3.1 A queue of jobs . . . . .	224
3.2 A loop that runs the next ready job . . . . .	224
3.3 Two senses of “ready”, and the generator that gives us pause . . . . .	224
3.4 The loop sleeps when nothing is ready . . . . .	224
3.5 Connecting the toy loop to news.py . . . . .	224
Summary . . . . .	225
<b>Coroutines, Without the Magic . . . . .</b>	<b>226</b>
4.1 Generators, the original coroutine . . . . .	226
4.2 yield as a pause point, one more time . . . . .	226
4.3 The toy loop . . . . .	226
4.4 What async def returns . . . . .	226
Polite driver for <code>x in gen():</code> (no for; the loop handles it) . . . . .	226
. . . . .	226
4.5 Coroutines are not running until the loop runs them . . . . .	227
Summary . . . . .	227
<b>async and await . . . . .</b>	<b>228</b>
5.1 What async does to a function definition . . . . .	228
5.2 What await does at a call site . . . . .	228
5.3 What can be awaited, and what cannot . . . . .	228
5.4 The first real asyncio news.py . . . . .	228
5.5 The cliffhanger . . . . .	228
5.6 The same keywords, on raw TCP . . . . .	229
Summary . . . . .	229
<b>await Yields, Suspends, and Resumes . . . . .</b>	<b>230</b>
6.1 What await does . . . . .	230
6.2 What is saved across the pause . . . . .	230
6.3 How the loop wakes the coroutine back up . . . . .	230
6.4 A Task is a coroutine that the loop is driving . . . . .	230
6.5 <i>For the curious:</i> <code>coro.send(value)</code> and the bytecode . . . . .	230
6.6 <i>For the curious:</i> selectors and the operating system . . . . .	230
6.7 Watching news.py yield and resume . . . . .	231
Summary . . . . .	231
<b>asyncio.run, create_task, gather . . . . .</b>	<b>232</b>

## CONTENTS

7.1 <code>asyncio.run</code> as the starting line . . . . .	232
7.2 What <code>asyncio.run</code> does . . . . .	232
7.3 <code>await coro</code> is sequential; <code>create_task(coro)</code> is concurrent . . . . .	232
7.4 <code>asyncio.gather</code> for “run these together” . . . . .	232
7.5 The fast news .py . . . . .	232
7.6 Concurrent TCP probes . . . . .	232
7.7 The lost-task trap . . . . .	233
Summary . . . . .	233
<b>The Time Toolbox . . . . .</b>	<b>234</b>
8.1 <code>asyncio.sleep</code> yields; <code>time.sleep</code> blocks . . . . .	234
8.2 <code>wait_for</code> for a deadline around an awaitable . . . . .	234
8.3 <code>asyncio.timeout()</code> as a context manager (Python 3.11+) . . . . .	234
8.4 <code>asyncio.shield</code> to protect from cancellation . . . . .	234
8.5 <code>asyncio.wait</code> vs <code>asyncio.gather</code> . . . . .	234
8.6 <code>asyncio.as_completed</code> for results as they arrive . . . . .	234
8.7 Timeouts and streaming . . . . .	235
Summary . . . . .	235
<b>async for, async with, Async Iteration and Resource Management . . . . .</b>	<b>236</b>
9.1 <code>async for</code> and the iterator protocol . . . . .	236
9.2 Async generators . . . . .	236
9.3 <code>async for</code> over <code>aiohttp</code> response bodies . . . . .	236
9.4 <code>async with</code> and the context-manager protocol . . . . .	236
9.5 Writing your own <code>async</code> context manager . . . . .	237
9.6 The streaming news .py . . . . .	237
Summary . . . . .	237
<b>Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars . . . . .</b>	<b>238</b>
10.1 <code>Task</code> is the loop’s unit of work; the coroutine is just the recipe . . . . .	238
10.2 <code>asyncio.TaskGroup</code> : structured concurrency by construction . . . . .	238
10.3 Cancellation: <code>task.cancel()</code> and <code>CancelledError</code> at the next <code>await</code> . . . . .	238
10.4 The bare-except trap . . . . .	238
10.5 <code>ExceptionGroup</code> and <code>except*</code> . . . . .	238
10.6 Re-raising <code>CancelledError</code> after cleanup . . . . .	239
10.7 Canceling a TCP probe . . . . .	239
10.8 Synchronization primitives . . . . .	239
10.9 Capping concurrency with <code>Semaphore</code> : 50 sites, 10 in flight . . . . .	240

## CONTENTS

10.10 contextvars for task-scoped state . . . . .	240
10.11 Producer-consumer with <code>asyncio.Queue</code> . . . . .	240
Summary . . . . .	240
<b>What Goes Wrong, and How to Find It . . . . .</b>	<b>241</b>
11.1 The failure mode: one synchronous call freezes the whole loop . . .	241
11.2 The blocking patterns to learn to recognize . . . . .	241
11.3 The single most useful tool: <code>asyncio</code> 's debug mode . . . . .	241
11.4 The four warnings the runtime emits . . . . .	241
11.5 <code>asyncio.all_tasks()</code> and <code>task.get_stack()</code> . . . . .	241
11.6 <code>aiomonitor</code> : a live REPL inside a running program . . . . .	241
11.7 Profiling that sees the loop correctly . . . . .	242
11.8 Logging task names . . . . .	242
11.9 <code>pdb</code> and <code>async</code> . . . . .	242
11.10 A guided debugging session on the <code>broken_news.py</code> . . . . .	242
11.11 <code>tcpcheck.py</code> as a diagnostic tool . . . . .	242
Summary . . . . .	242
<b>Testing async code . . . . .</b>	<b>243</b>
12.1 Your first async test . . . . .	243
12.2 The standard-library alternative . . . . .	243
12.3 What mocks are, and why async needs a different one . . . . .	243
12.4 Testing cancellation . . . . .	243
12.5 <i>Optional</i> : Testing <code>TaskGroup</code> error propagation . . . . .	243
12.6 Event loop fixture scope . . . . .	243
12.7 Testing timeouts without real time . . . . .	244
12.8 The most subtle async test bug . . . . .	244
12.9 When a test hangs . . . . .	244
12.10 The full test suite for the <code>fetcher</code> . . . . .	244
Summary . . . . .	244
<b>Where <code>asyncio</code> Ends: Executors, Graceful Shutdown . . . . .</b>	<b>245</b>
13.1 Fixing the parser bug from Chapter 11 . . . . .	245
13.2 The executor toolbox . . . . .	245
13.3 The decision tree . . . . .	245
Heavy CPU work that needs to scale across cores Use <code>multiprocess-</code> <code>ing</code> directly. <code>asyncio</code> is the wrong tool for that workload. . . . .	245
13.4 Bridging from sync to async, and back . . . . .	245
13.5 Library design: expose async, do not hide it . . . . .	246

CONTENTS

13.6 Graceful shutdown . . . . .	246
13.7 The final news.py . . . . .	246
Summary . . . . .	246
<b>Appendix A: Setting up your environment . . . . .</b>	<b>248</b>
A.1 Python versions and what they buy you . . . . .	248
A.2 The packages the book uses . . . . .	248
A.3 The smallest runnable asyncio script . . . . .	248
A.4 Reading the running example as you go . . . . .	248
<b>Appendix B: The asyncio API surface . . . . .</b>	<b>249</b>
Functions . . . . .	249
<code>asyncio.wait_for(coro, timeout)</code> Wraps a single awaitable with a deadline; cancels the inner work and raises <code>TimeoutError</code> on expiry. 8 . . . . .	249
Classes and protocols . . . . .	249
<code>ExceptionGroup</code> The exception type that holds multiple exceptions raised by sibling <code>Tasks</code> in a <code>TaskGroup</code> . Caught with <code>except*</code> (3.11+). 10 . . . . .	249
Loop methods worth knowing . . . . .	249
<code>loop.set_default_executor(executor)</code> Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13 . . . . .	250
<code>aiohttp</code> API surface used in the book . . . . .	250
<code>response.text()</code> Awaitable that returns the full response body as a decoded string. 5 . . . . .	250
Where to find the rest . . . . .	250
<b>Appendix C: Troubleshooting guide . . . . .</b>	<b>251</b>
C.1 Asyncio runtime warnings, what they mean and what to fix . . . . .	251
<code>RuntimeWarning: async generator 'X' was never closed</code> An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling <code>aclose()</code> . Wrap resources held between yields in <code>try/finally</code> . Call <code>aclose()</code> explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.) . . . . .	251
C.2 The script runs without errors but is no faster than synchronous .	251

CONTENTS

Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.) . . . . . 251

C.3 The program hangs or never exits . . . . . 252

Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.) . . . . . 252

C.4 Common errors and what they mean . . . . . 252

RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup. . . . . 253

When the table does not have your symptom . . . . . 253

**Appendix D: Further reading . . . . . 254**

D.1 The asyncio source code . . . . . 254

D.2 The async library ecosystem . . . . . 254

D.3 The structured concurrency literature . . . . . 254

D.4 The PEPs that shaped asyncio . . . . . 254

D.5 Things this book did not cover, that you may want next . . . . . 254

**Glossary . . . . . 255**

**About the Author . . . . . 256**

**From the Author . . . . . 257**

**Preface . . . . . 258**

## CONTENTS

<b>What This Book Is About</b> . . . . .	<b>259</b>
<b>What This Book Is Not</b> . . . . .	<b>260</b>
<b>Who This Book Is For</b> . . . . .	<b>261</b>
<b>How to Use This Book</b> . . . . .	<b>262</b>
<b>Conventions Used in This Book</b> . . . . .	<b>263</b>
<b>Companion Code and Data</b> . . . . .	<b>264</b>
<b>How to Contact Us</b> . . . . .	<b>265</b>
<b>Acknowledgments</b> . . . . .	<b>266</b>
<b>Contents</b> . . . . .	<b>267</b>
<b>The Blocking Function</b> . . . . .	<b>268</b>
1.1 The first version of news.py . . . . .	268
1.2 What “blocking” actually means . . . . .	268
1.3 The cost grows with the list . . . . .	268
1.4 The same shape, one layer down . . . . .	268
1.5 The question that drives the rest of the book . . . . .	268
Summary . . . . .	268
<b>Concurrency, Parallelism, and the GIL</b> . . . . .	<b>270</b>
2.1 What a thread is, briefly . . . . .	270
2.2 The threaded version of news.py . . . . .	270
2.3 Threading the TCP probes . . . . .	270
2.4 Why the speedup actually works (and the GIL story you have heard wrong) . . . . .	270
2.5 Why threads are not the answer the book is reaching for . . . . .	270
2.6 The two scales the book cares about . . . . .	270
2.7 Where the next chapter goes . . . . .	271
Summary . . . . .	271
<b>The Loop, From Scratch</b> . . . . .	<b>272</b>
3.1 A queue of jobs . . . . .	272
3.2 A loop that runs the next ready job . . . . .	272
3.3 Two senses of “ready”, and the generator that gives us pause . . . . .	272
3.4 The loop sleeps when nothing is ready . . . . .	272

CONTENTS

3.5 Connecting the toy loop to news.py . . . . .	272
Summary . . . . .	273
<b>Coroutines, Without the Magic . . . . .</b>	<b>274</b>
4.1 Generators, the original coroutine . . . . .	274
4.2 yield as a pause point, one more time . . . . .	274
4.3 The toy loop . . . . .	274
4.4 What async def returns . . . . .	274
Polite driver for <code>x in gen():</code> (no <code>for</code> ; the loop handles it) . . . . .	274
. . . . .	274
4.5 Coroutines are not running until the loop runs them . . . . .	275
Summary . . . . .	275
<b>async and await . . . . .</b>	<b>276</b>
5.1 What async does to a function definition . . . . .	276
5.2 What await does at a call site . . . . .	276
5.3 What can be awaited, and what cannot . . . . .	276
5.4 The first real <code>asyncio</code> news.py . . . . .	276
5.5 The cliffhanger . . . . .	276
5.6 The same keywords, on raw TCP . . . . .	277
Summary . . . . .	277
<b>await Yields, Suspends, and Resumes . . . . .</b>	<b>278</b>
6.1 What await does . . . . .	278
6.2 What is saved across the pause . . . . .	278
6.3 How the loop wakes the coroutine back up . . . . .	278
6.4 A <code>Task</code> is a coroutine that the loop is driving . . . . .	278
6.5 <i>For the curious:</i> <code>coro.send(value)</code> and the bytecode . . . . .	278
6.6 <i>For the curious:</i> selectors and the operating system . . . . .	278
6.7 Watching news.py yield and resume . . . . .	279
Summary . . . . .	279
<b>asyncio.run, create_task, gather . . . . .</b>	<b>280</b>
7.1 <code>asyncio.run</code> as the starting line . . . . .	280
7.2 What <code>asyncio.run</code> does . . . . .	280
7.3 <code>await coro</code> is sequential; <code>create_task(coro)</code> is concurrent . . . . .	280
7.4 <code>asyncio.gather</code> for “run these together” . . . . .	280
7.5 The fast news.py . . . . .	280
7.6 Concurrent TCP probes . . . . .	280
7.7 The lost-task trap . . . . .	281

Summary . . . . .	281
<b>The Time Toolbox . . . . .</b>	<b>282</b>
8.1 <code>asyncio.sleep</code> yields; <code>time.sleep</code> blocks . . . . .	282
8.2 <code>wait_for</code> for a deadline around an awaitable . . . . .	282
8.3 <code>asyncio.timeout()</code> as a context manager (Python 3.11+) . . . . .	282
8.4 <code>asyncio.shield</code> to protect from cancellation . . . . .	282
8.5 <code>asyncio.wait</code> vs <code>asyncio.gather</code> . . . . .	282
8.6 <code>asyncio.as_completed</code> for results as they arrive . . . . .	282
8.7 Timeouts and streaming . . . . .	283
Summary . . . . .	283
<b>async for, async with, Async Iteration and Resource Management . . . . .</b>	<b>284</b>
9.1 <code>async for</code> and the iterator protocol . . . . .	284
9.2 Async generators . . . . .	284
9.3 <code>async for</code> over <code>aiorttp</code> response bodies . . . . .	284
9.4 <code>async with</code> and the context-manager protocol . . . . .	284
9.5 Writing your own <code>async</code> context manager . . . . .	285
9.6 The streaming <code>news.py</code> . . . . .	285
Summary . . . . .	285
<b>Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars . . . . .</b>	<b>286</b>
10.1 <code>Task</code> is the loop's unit of work; the coroutine is just the recipe . . . . .	286
10.2 <code>asyncio.TaskGroup</code> : structured concurrency by construction . . . . .	286
10.3 Cancellation: <code>task.cancel()</code> and <code>CancelledError</code> at the next <code>await</code> . . . . .	286
10.4 The bare- <code>except</code> trap . . . . .	286
10.5 <code>ExceptionGroup</code> and <code>except*</code> . . . . .	286
10.6 Re-raising <code>CancelledError</code> after cleanup . . . . .	287
10.7 Canceling a TCP probe . . . . .	287
10.8 Synchronization primitives . . . . .	287
10.9 Capping concurrency with <code>Semaphore</code> : 50 sites, 10 in flight . . . . .	288
10.10 <code>contextvars</code> for task-scoped state . . . . .	288
10.11 Producer-consumer with <code>asyncio.Queue</code> . . . . .	288
Summary . . . . .	288
<b>What Goes Wrong, and How to Find It . . . . .</b>	<b>289</b>
11.1 The failure mode: one synchronous call freezes the whole loop . . . . .	289
11.2 The blocking patterns to learn to recognize . . . . .	289
11.3 The single most useful tool: <code>asyncio</code> 's debug mode . . . . .	289

## CONTENTS

11.4 The four warnings the runtime emits . . . . .	289
11.5 <code>asyncio.all_tasks()</code> and <code>task.get_stack()</code> . . . . .	289
11.6 <code>aiomonitor</code> : a live REPL inside a running program . . . . .	289
11.7 Profiling that sees the loop correctly . . . . .	290
11.8 Logging task names . . . . .	290
11.9 <code>pdb</code> and <code>async</code> . . . . .	290
11.10 A guided debugging session on the <code>broken_news.py</code> . . . . .	290
11.11 <code>tcpcheck.py</code> as a diagnostic tool . . . . .	290
Summary . . . . .	290
<b>Testing async code . . . . .</b>	<b>291</b>
12.1 Your first async test . . . . .	291
12.2 The standard-library alternative . . . . .	291
12.3 What mocks are, and why async needs a different one . . . . .	291
12.4 Testing cancellation . . . . .	291
12.5 <i>Optional</i> : Testing <code>TaskGroup</code> error propagation . . . . .	291
12.6 Event loop fixture scope . . . . .	291
12.7 Testing timeouts without real time . . . . .	292
12.8 The most subtle async test bug . . . . .	292
12.9 When a test hangs . . . . .	292
12.10 The full test suite for the <code>fetcher</code> . . . . .	292
Summary . . . . .	292
<b>Where asyncio Ends: Executors, Graceful Shutdown . . . . .</b>	<b>293</b>
13.1 Fixing the parser bug from Chapter 11 . . . . .	293
13.2 The executor toolbox . . . . .	293
13.3 The decision tree . . . . .	293
Heavy CPU work that needs to scale across cores Use <code>multiprocess-</code> <code>ing</code> directly. <code>asyncio</code> is the wrong tool for that workload. . . . .	293
13.4 Bridging from sync to async, and back . . . . .	293
13.5 Library design: expose async, do not hide it . . . . .	294
13.6 Graceful shutdown . . . . .	294
13.7 The final <code>news.py</code> . . . . .	294
Summary . . . . .	294
<b>Appendix A: Setting up your environment . . . . .</b>	<b>296</b>
A.1 Python versions and what they buy you . . . . .	296
A.2 The packages the book uses . . . . .	296
A.3 The smallest runnable <code>asyncio</code> script . . . . .	296

A.4 Reading the running example as you go . . . . .	296
<b>Appendix B: The asyncio API surface . . . . .</b>	<b>297</b>
Functions . . . . .	297
<code>asyncio.wait_for(coro, timeout)</code> Wraps a single awaitable with a deadline; cancels the inner work and raises <code>TimeoutError</code> on expiry. 8 . . . . .	297
Classes and protocols . . . . .	297
<code>ExceptionGroup</code> The exception type that holds multiple exceptions raised by sibling <code>Tasks</code> in a <code>TaskGroup</code> . Caught with <code>except*</code> (3.11+). 10 . . . . .	297
Loop methods worth knowing . . . . .	297
<code>loop.set_default_executor(executor)</code> Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13 . . . . .	298
aiohttp API surface used in the book . . . . .	298
<code>response.text()</code> Awaitable that returns the full response body as a decoded string. 5 . . . . .	298
Where to find the rest . . . . .	298
<b>Appendix C: Troubleshooting guide . . . . .</b>	<b>299</b>
C.1 Asyncio runtime warnings, what they mean and what to fix . . . . .	299
<code>RuntimeWarning: async generator 'X' was never closed</code> An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling <code>aclose()</code> . Wrap resources held between yields in <code>try/finally</code> . Call <code>aclose()</code> explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1) . . . . .	299
C.2 The script runs without errors but is no faster than synchronous .	299
Slow-callback warnings fire ( <code>Executing &lt;Task ...&gt; took 0.234</code> seconds) when running with <code>debug=True</code> . A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with <code>asyncio.to_thread</code> . For CPU-bound work that needs cores, use a process pool via <code>loop.run_in_executor(ProcessPoolExecutor(), ...)</code> . (Chapter 11; Chapter 13.) . . . . .	299
C.3 The program hangs or never exits . . . . .	300

CONTENTS

Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.) . . . . . 300

C.4 Common errors and what they mean . . . . . 300

RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup. . . . . 301

When the table does not have your symptom . . . . . 301

**Appendix D: Further reading . . . . . 302**

D.1 The asyncio source code . . . . . 302

D.2 The async library ecosystem . . . . . 302

D.3 The structured concurrency literature . . . . . 302

D.4 The PEPs that shaped asyncio . . . . . 302

D.5 Things this book did not cover, that you may want next . . . . . 302

**Glossary . . . . . 303**

**About the Author . . . . . 304**

**From the Author . . . . . 305**

**Preface . . . . . 306**

**What This Book Is About . . . . . 307**

**What This Book Is Not . . . . . 308**

**Who This Book Is For . . . . . 309**

**How to Use This Book . . . . . 310**

**Conventions Used in This Book . . . . . 311**

**Companion Code and Data . . . . . 312**

CONTENTS

<b>How to Contact Us</b> . . . . .	<b>313</b>
<b>Acknowledgments</b> . . . . .	<b>314</b>
<b>Contents</b> . . . . .	<b>315</b>
<b>The Blocking Function</b> . . . . .	<b>316</b>
1.1 The first version of <code>news.py</code> . . . . .	316
1.2 What “blocking” actually means . . . . .	316
1.3 The cost grows with the list . . . . .	316
1.4 The same shape, one layer down . . . . .	316
1.5 The question that drives the rest of the book . . . . .	316
Summary . . . . .	316
<b>Concurrency, Parallelism, and the GIL</b> . . . . .	<b>318</b>
2.1 What a thread is, briefly . . . . .	318
2.2 The threaded version of <code>news.py</code> . . . . .	318
2.3 Threading the TCP probes . . . . .	318
2.4 Why the speedup actually works (and the GIL story you have heard wrong) . . . . .	318
2.5 Why threads are not the answer the book is reaching for . . . . .	318
2.6 The two scales the book cares about . . . . .	318
2.7 Where the next chapter goes . . . . .	319
Summary . . . . .	319
<b>The Loop, From Scratch</b> . . . . .	<b>320</b>
3.1 A queue of jobs . . . . .	320
3.2 A loop that runs the next ready job . . . . .	320
3.3 Two senses of “ready”, and the generator that gives us pause . . . . .	320
3.4 The loop sleeps when nothing is ready . . . . .	320
3.5 Connecting the toy loop to <code>news.py</code> . . . . .	320
Summary . . . . .	321
<b>Coroutines, Without the Magic</b> . . . . .	<b>322</b>
4.1 Generators, the original coroutine . . . . .	322
4.2 <code>yield</code> as a pause point, one more time . . . . .	322
4.3 The toy loop . . . . .	322
4.4 What <code>async def</code> returns . . . . .	322
Polite driver for <code>x in gen()</code> : (no <code>for</code> ; the loop handles it) . . . . .	322
. . . . .	322

## CONTENTS

4.5 Coroutines are not running until the loop runs them . . . . .	323
Summary . . . . .	323
<b>async and await . . . . .</b>	<b>324</b>
5.1 What async does to a function definition . . . . .	324
5.2 What await does at a call site . . . . .	324
5.3 What can be awaited, and what cannot . . . . .	324
5.4 The first real <code>asyncio</code> news.py . . . . .	324
5.5 The cliffhanger . . . . .	324
5.6 The same keywords, on raw TCP . . . . .	325
Summary . . . . .	325
<b>await Yields, Suspends, and Resumes . . . . .</b>	<b>326</b>
6.1 What await does . . . . .	326
6.2 What is saved across the pause . . . . .	326
6.3 How the loop wakes the coroutine back up . . . . .	326
6.4 A Task is a coroutine that the loop is driving . . . . .	326
6.5 <i>For the curious</i> : <code>coro.send(value)</code> and the bytecode . . . . .	326
6.6 <i>For the curious</i> : selectors and the operating system . . . . .	326
6.7 Watching news.py yield and resume . . . . .	327
Summary . . . . .	327
<b>asyncio.run, create_task, gather . . . . .</b>	<b>328</b>
7.1 <code>asyncio.run</code> as the starting line . . . . .	328
7.2 What <code>asyncio.run</code> does . . . . .	328
7.3 <code>await coro</code> is sequential; <code>create_task(coro)</code> is concurrent . . . . .	328
7.4 <code>asyncio.gather</code> for “run these together” . . . . .	328
7.5 The fast news.py . . . . .	328
7.6 Concurrent TCP probes . . . . .	328
7.7 The lost-task trap . . . . .	329
Summary . . . . .	329
<b>The Time Toolbox . . . . .</b>	<b>330</b>
8.1 <code>asyncio.sleep</code> yields; <code>time.sleep</code> blocks . . . . .	330
8.2 <code>wait_for</code> for a deadline around an awaitable . . . . .	330
8.3 <code>asyncio.timeout()</code> as a context manager (Python 3.11+) . . . . .	330
8.4 <code>asyncio.shield</code> to protect from cancellation . . . . .	330
8.5 <code>asyncio.wait</code> vs <code>asyncio.gather</code> . . . . .	330
8.6 <code>asyncio.as_completed</code> for results as they arrive . . . . .	330
8.7 Timeouts and streaming . . . . .	331

## CONTENTS

Summary . . . . .	331
<b>async for, async with, Async Iteration and Resource Management . . .</b>	<b>332</b>
9.1 async for and the iterator protocol . . . . .	332
9.2 Async generators . . . . .	332
9.3 async for over aiohttp response bodies . . . . .	332
9.4 async with and the context-manager protocol . . . . .	332
9.5 Writing your own async context manager . . . . .	333
9.6 The streaming news.py . . . . .	333
Summary . . . . .	333
<b>Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars .</b>	<b>334</b>
10.1 Task is the loop's unit of work; the coroutine is just the recipe . .	334
10.2 asyncio.TaskGroup: structured concurrency by construction . . .	334
10.3 Cancellation: task.cancel() and CancelledError at the next await . . . . .	334
10.4 The bare-except trap . . . . .	334
10.5 ExceptionGroup and except* . . . . .	334
10.6 Re-raising CancelledError after cleanup . . . . .	335
10.7 Canceling a TCP probe . . . . .	335
10.8 Synchronization primitives . . . . .	335
10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight . . . .	336
10.10 contextvars for task-scoped state . . . . .	336
10.11 Producer-consumer with asyncio.Queue . . . . .	336
Summary . . . . .	336
<b>What Goes Wrong, and How to Find It . . . . .</b>	<b>337</b>
11.1 The failure mode: one synchronous call freezes the whole loop . .	337
11.2 The blocking patterns to learn to recognize . . . . .	337
11.3 The single most useful tool: asyncio's debug mode . . . . .	337
11.4 The four warnings the runtime emits . . . . .	337
11.5 asyncio.all_tasks() and task.get_stack() . . . . .	337
11.6 aiomonitor: a live REPL inside a running program . . . . .	337
11.7 Profiling that sees the loop correctly . . . . .	338
11.8 Logging task names . . . . .	338
11.9 pdb and async . . . . .	338
11.10 A guided debugging session on the broken news.py . . . . .	338
11.11 tcpcheck.py as a diagnostic tool . . . . .	338
Summary . . . . .	338

<b>Testing async code</b>	<b>339</b>
12.1 Your first async test	339
12.2 The standard-library alternative	339
12.3 What mocks are, and why async needs a different one	339
12.4 Testing cancellation	339
12.5 <i>Optional</i> : Testing TaskGroup error propagation	339
12.6 Event loop fixture scope	339
12.7 Testing timeouts without real time	340
12.8 The most subtle async test bug	340
12.9 When a test hangs	340
12.10 The full test suite for the fetcher	340
Summary	340
<b>Where asyncio Ends: Executors, Graceful Shutdown</b>	<b>341</b>
13.1 Fixing the parser bug from Chapter 11	341
13.2 The executor toolbox	341
13.3 The decision tree	341
Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.	341
13.4 Bridging from sync to async, and back	341
13.5 Library design: expose async, do not hide it	342
13.6 Graceful shutdown	342
13.7 The final news.py	342
Summary	342
<b>Appendix A: Setting up your environment</b>	<b>344</b>
A.1 Python versions and what they buy you	344
A.2 The packages the book uses	344
A.3 The smallest runnable asyncio script	344
A.4 Reading the running example as you go	344
<b>Appendix B: The asyncio API surface</b>	<b>345</b>
Functions	345
<code>asyncio.wait_for(coro, timeout)</code> Wraps a single awaitable with a deadline; cancels the inner work and raises <code>TimeoutError</code> on expiry. 8	345
Classes and protocols	345

CONTENTS

ExceptionGroup The exception type that holds multiple exceptions raised by sibling Tasks in a TaskGroup. Caught with `except*` (3.11+). 10 . . . . . 345

Loop methods worth knowing . . . . . 345

`loop.set_default_executor(executor)` Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13 . . . . . 346

aiohhttp API surface used in the book . . . . . 346

`response.text()` Awaitable that returns the full response body as a decoded string. 5 . . . . . 346

Where to find the rest . . . . . 346

**Appendix C: Troubleshooting guide . . . . . 347**

C.1 Asyncio runtime warnings, what they mean and what to fix . . . . 347

RuntimeWarning: async generator 'X' was never closed An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between yields in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.) . . . . . 347

C.2 The script runs without errors but is no faster than synchronous . 347

Slow-callback warnings fire (`Executing <Task ...> took 0.234 seconds`) when running with `debug=True`. A coroutine ran for too long between `awaits`, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.) . . . . . 347

C.3 The program hangs or never exits . . . . . 348

Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.) . . . . . 348

C.4 Common errors and what they mean . . . . . 348

<p>RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after <code>asyncio.run</code> returned and closed it. Common when a finalizer or <code>__del__</code> method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (<code>finally</code> blocks, <code>__aexit__</code>). Do not rely on <code>__del__</code> or <code>atexit</code> for async cleanup. . . . .</p> <p>When the table does not have your symptom . . . . .</p>	<p>349</p> <p>349</p>
<b>Appendix D: Further reading . . . . .</b>	<b>350</b>
D.1 The asyncio source code . . . . .	350
D.2 The async library ecosystem . . . . .	350
D.3 The structured concurrency literature . . . . .	350
D.4 The PEPs that shaped asyncio . . . . .	350
D.5 Things this book did not cover, that you may want next . . . . .	350
<b>Glossary . . . . .</b>	<b>351</b>
<b>About the Author . . . . .</b>	<b>352</b>

# Getting Started

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Unsure How to Get Started? Try our Book Workshop!

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## How to Write on Leanpub

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Previewing and publishing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Basic formatting

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Markdown and Markua

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Generate a preview version of your book**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Either read a tutorial, or just go for it!**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Read the tutorial...**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **...or just go for it!**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Thanks for being a Leanpub author!**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Writing in Markua

Writing in Markua is easy! You can learn most of what you need to know with just a few examples.

To make *italic text* you surround it with single asterisks. To make **bold text** you surround it with double asterisks.

## Section One

You can start new sections by starting a line with two # signs and a space, and then typing your section title.

### Sub-Section One

You can start new sub-sections by starting a line with three # signs and a space, and then typing your sub-section title.

## Including a Chapter in the Sample Book

At the top of this file, you will also see a line at the top:

```
1 {sample: true}
```

Leanpub has the ability to make a sample book, which interested readers can download or read online. If you add this line above a chapter heading, then when you publish your book, this chapter will be included in a separate sample book for these interested readers.

## Links

You can add web links easily.

Here's a link to the [Leanpub homepage](#).

## Images

You can add an image to your book in a similar way.

First, add the image to the “Resources” folder for your book. You will find the “Resources” folder under the “Manuscript” menu to the left.

If you look in your book’s “Resources” folder right now, you will see that there is an example image there with the file name “palm-trees.jpg”. Here’s how you can add this image to your book:



If you want to add a figure title, you put it in quotes:



**Figure 1. Palm Trees**

If you want to add descriptive alt text, which is good for accessibility, you put it between the square brackets:



**Figure 2. Palm Trees**

You can also set the alt text and/or the figure title in an attribute list:



**Figure 3. Palm Trees**

Finally, if no title is provided, and the `alt-title` document setting is the default of `all`, the alt text will be used as the figure title instead of as alt text.



**Figure 4. Palm Trees**

You can set the important document settings at Settings > Generation Settings.

## **Lists**

### **Numbered Lists**

You make a numbered list like this:

1. kale
2. carrot
3. ginger

### **Bulleted Lists**

You make a bulleted list like this:

- kale

- carrot
- ginger

## Definition Lists

You can even have definition lists!

### **term 1**

definition 1a

definition 1b

### **term 2**

definition 2

## Page Breaks

We don't recommend that you manually break pages, since that is brittle and can lead to unexpected formatting if you edit text earlier in your chapter and forget about the manual page breaks. But if you really want to add a page break, you use the `{pagebreak}` directive on a line by itself, with blank lines above it and below it.

## Code Samples

You can add code samples really easily. Code can be in separate files (a “local” resource) or in the manuscript itself (an “inline” resource).

### Local Code Samples

Here’s a local code resource:

**Figure 5. Hello World in Ruby**

---

```
1 require 'time'
2
3 # This is just some pointless code so you can see the syntax highlighting...
4 def display_info
5   pi = Math::PI.round(10)
6   time_last_year = (Time.now - 365 * 24 * 60 * 60).getlocal("-08:00")
7   formatted_time = time_last_year.strftime("%Y-%m-%d %H:%M:%S")
8   puts "Pi to 10 decimal places: #{pi}"
9   puts "The time 1 year ago in Pacific Time: #{formatted_time}"
10 end
```

---

### Inline Code Samples

Inline code samples can either be spans or figures.

A span looks like `puts "hello world"` this.

A figure looks like this:

```
1 require 'time'
2
3 # This is just some pointless code so you can see the syntax highlighting...
4 def display_info
5   pi = Math::PI.round(10)
6   time_last_year = (Time.now - 365 * 24 * 60 * 60).getlocal("-08:00")
7   formatted_time = time_last_year.strftime("%Y-%m-%d %H:%M:%S")
8   puts "Pi to 10 decimal places: #{pi}"
9   puts "The time 1 year ago in Pacific Time: #{formatted_time}"
10 end
```

You can also add a figure title using the title attribute:

**Figure 6. Hello World in Ruby**


---

```

1 require 'time'
2
3 # This is just some pointless code so you can see the syntax highlighting...
4 def display_info
5   pi = Math::PI.round(10)
6   time_last_year = (Time.now - 365 * 24 * 60 * 60).getlocal("-08:00")
7   formatted_time = time_last_year.strftime("%Y-%m-%d %H:%M:%S")
8   puts "Pi to 10 decimal places: #{pi}"
9   puts "The time 1 year ago in Pacific Time: #{formatted_time}"
10 end

```

---

## Tables

You can insert tables easily inline, using the GitHub Flavored Markdown (GFM) table syntax:

Header 1	Header 2
Content 1	Content 2
Content 3	Content 4 Can be Different Length

Tables work best for numeric tabular data involving a small number of columns containing small numbers:

Central Bank	Rate
JPY	-0.10%
EUR	0.00%
USD	0.00%
CAD	0.25%

Definition lists are preferred to tables for most use cases, since reading a large table with many columns is terrible on phones and since typing text in a table quickly gets annoying.

## Math

You can easily insert math equations inline using either spans or figures.

Here's one of the kinematic equations  $d = v_i t + \frac{1}{2} a t^2$  inserted as a span inside a sentence.

Here's some math inserted as a figure.

$$\left| \sum_{i=1}^n a_i b_i \right| \leq \left( \sum_{i=1}^n a_i^2 \right)^{1/2} \left( \sum_{i=1}^n b_i^2 \right)^{1/2}$$

**Figure 7. Something Involving Sums**

## Headings

Markua supports both of Markdown's heading styles.

The preferred style, called atx headers, has the following meaning in Markua:

```

1 {class: part}
2 # Part
3
4 This is a paragraph.
5
6 # Chapter
7
8 This is a paragraph.
9
10 ## Section
11
12 This is a paragraph.
13
14 ### Sub-section
15
16 This is a paragraph.
17
18 #### Sub-sub-section
19
20 This is a paragraph.
21
22 ##### Sub-sub-sub-section
23
```

```

24 This is a paragraph.
25
26 ##### Sub-sub-sub-sub-section
27
28 This is a paragraph.

```

Note the use of three backticks in the above example, to treat the Markua like inline code (instead of actually like headers).

The other style of headers, called Setext headers, has the following headings:

```

1 {class: part}
2 Part
3 ====
4
5 This is a paragraph.
6
7 Chapter
8 =====
9
10 This is a paragraph.
11
12 Section
13 -----
14
15 This is a paragraph.

```

Setext headers look nice, but only if you're only using chapters and sections. If you want to add sub-sections (or lower), you'll be using atx headers for at least some of your headers. My advice is to just use atx headers all the time. (The `{class: part}` attribute list on a chapter header to make a part header does actually work with Setext headers, but it's really ugly.)

Note that while it is confusing and ugly to mix and match using atx and Setext headers for chapters and sections in the same document, you can do it. However, please don't.

## Block quotes, Asides and Blurbs

Block quotes are really easy too.

—Peter Armstrong, *Markua Spec*

Asides are useful for longer text.  
But typing them like this isn't fun.

Asides can be written this way, since adding a bunch of A> stuff at the beginning of each line can get annoying with longer asides.

Blurbs are useful

Blurbs are useful

There are many types of blurbs, which will be familiar to you if you've ever read a computer programming book.



This is a discussion.

You can also specify them this way:



This is a discussion



This is an error.



This is information.



This is a question. (Not a question in a Markua course; those are done differently!)



This is a tip.



This is a warning.



This is an exercise. (Not an exercise in a Markua course; those are done differently!)

## Good luck, have fun!

If you've read this far, you're definitely the right type of person to be here!

Our last piece of advice is simple: once you have a couple chapters completed, publish your book in-progress!

This approach is called Lean Publishing. It's why Leanpub is called Leanpub.

If you want to learn more about Lean Publishing, read [this](#) or watch [this](#).

Asyncio from ground up

A Working Mental Model for Python asyncio

Ritesh Modi

Asyncio from ground up: A Working Mental Model for Python asyncio

First Edition

Copyright © 2026 Ritesh Modi. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the

copyright holder, except in the case of brief quotations embodied in critical reviews and certain other noncommercial uses permitted by copyright law.

Product names, logos, and brands mentioned in this book are the property of their respective owners. Use of a trademark in this book does not imply endorsement by the trademark holder. Python is a trademark of the Python Software Foundation.

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 book, neither the author nor the publisher 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 or code examples contained herein.

Code examples in this book are released under the MIT License and may be used freely in your own projects, commercial or otherwise, with or without attribution.

First printing: 2026

For everyone who closed three asyncio tutorials in a row and thought it was their fault.

# From the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Preface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is About

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is Not

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Who This Book Is For

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Use This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Conventions Used in This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Companion Code and Data

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Contact Us

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Acknowledgments

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Contents

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Blocking Function

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.1 The first version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.2 What “blocking” actually means

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.3 The cost grows with the list

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.4 The same shape, one layer down

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.5 The question that drives the rest of the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Concurrency, Parallelism, and the GIL

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.1 What a thread is, briefly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.2 The threaded version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.3 Threading the TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.4 Why the speedup actually works (and the GIL story you have heard wrong)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.5 Why threads are not the answer the book is reaching for

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.6 The two scales the book cares about

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.7 Where the next chapter goes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Loop, From Scratch

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.1 A queue of jobs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.2 A loop that runs the next ready job

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.3 Two senses of “ready”, and the generator that gives us pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.4 The loop sleeps when nothing is ready

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.5 Connecting the toy loop to news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **3.5.1 Common mistakes you will make at least once**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Coroutines, Without the Magic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.1 Generators, the original coroutine

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.2 yield as a pause point, one more time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.3 The toy loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.4 What async def returns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Polite driver for `x in gen()`: (no for; the loop handles it)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.5 Coroutines are not running until the loop runs them

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.1 Two coroutines yielding back and forth

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.2 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### Optional: how the bridge works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async and await

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.1 What async does to a function definition

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.2 What await does at a call site

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.3 What can be awaited, and what cannot

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.4 The first real asyncio news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 5.4.1 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.5 The cliffhanger

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.6 The same keywords, on raw TCP

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# await Yields, Suspends, and Resumes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.1 What await does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.2 What is saved across the pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.3 How the loop wakes the coroutine back up

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.4 A Task is a coroutine that the loop is driving

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.5 *For the curious: coro.send(value) and the bytecode*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.6 *For the curious*: selectors and the operating system

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.7 Watching news.py yield and resume

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# asyncio.run, create\_task, gather

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.1 asyncio.run as the starting line

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.2 What asyncio.run does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.3 await coro is sequential; create\_task(coro) is concurrent

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.4 asyncio.gather for “run these together”

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.5 The fast news .py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.6 Concurrent TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.7 The lost-task trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Time Toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.1 `asyncio.sleep` yields; `time.sleep` blocks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.2 `wait_for` for a deadline around an awaitable

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.3 `asyncio.timeout()` as a context manager (Python 3.11+)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.4 `asyncio.shield` to protect from cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.5 `asyncio.wait` vs `asyncio.gather`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.6 `asyncio.as_completed` for results as they arrive

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.7 Timeouts and streaming

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async for, async with, Async Iteration and Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.1 async for and the iterator protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.2 Async generators

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.2.1 Cleanup when iteration stops short

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.3 async for over aiohttp response bodies

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.3.1 Async comprehensions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.4 async with and the context-manager protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.4.1 Multiple async with and AsyncExitStack

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.5 Writing your own async context manager

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.6 The streaming news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.6.1 Two more async-generator pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.1 Task is the loop's unit of work; the coroutine is just the recipe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.2 asyncio.TaskGroup: structured concurrency by construction

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.3 Cancellation: `task.cancel()` and `CancelledError` at the next `await`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.4 The bare-except trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.5 ExceptionGroup and except\*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.6 Re-raising CancelledError after cleanup

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.7 Canceling a TCP probe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.8 Synchronization primitives

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Lock: only one at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Semaphore: only N at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Event: one coroutine signals, others wait**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **asyncio.Queue: pass items between coroutines**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.10 contextvars for task-scoped state**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.11 Producer-consumer with asyncio.Queue**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What Goes Wrong, and How to Find It

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.1 The failure mode: one synchronous call freezes the whole loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.2 The blocking patterns to learn to recognize

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.3 The single most useful tool: asyncio's debug mode

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.4 The four warnings the runtime emits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.5 `asyncio.all_tasks()` and `task.get_stack()`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.6 aiomonitor: a live REPL inside a running program

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.7 Profiling that sees the loop correctly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.8 Logging task names

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.9 pdb and async

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.10 A guided debugging session on the broken news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.11 tcpcheck.py as a diagnostic tool

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Testing async code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.1 Your first async test

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.2 The standard-library alternative

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.3 What mocks are, and why async needs a different one

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.4 Testing cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.5 *Optional*: Testing TaskGroup error propagation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.6 Event loop fixture scope

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.7 Testing timeouts without real time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.8 The most subtle async test bug

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.9 When a test hangs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.10 The full test suite for the fetcher

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Where asyncio Ends: Executors, Graceful Shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.1 Fixing the parser bug from Chapter 11

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.2 The executor toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.3 The decision tree

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.4 Bridging from sync to async, and back

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.5 Library design: expose async, do not hide it

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.6 Graceful shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.7 The final news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Closing reflection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Chapter recap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix A: Setting up your environment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.1 Python versions and what they buy you

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.2 The packages the book uses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.3 The smallest runnable asyncio script

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.4 Reading the running example as you go

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix B: The asyncio API surface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Functions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Classes and protocols

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except*` (3.11+). 10**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Loop methods worth knowing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **loop.set\_default\_executor(executor)** Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## aihttp API surface used in the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **response.text()** Awaitable that returns the full response body as a decoded string. 5

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Where to find the rest

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix C: Troubleshooting guide

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.1 Asyncio runtime warnings, what they mean and what to fix

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeWarning: async generator 'X' was never closed** An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between `yields` in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.2 The script runs without errors but is no faster than synchronous

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **C.3 The program hangs or never exits**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.4 Common errors and what they mean

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### When the table does not have your symptom

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix D: Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.1 The asyncio source code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.2 The async library ecosystem

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.3 The structured concurrency literature

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.4 The PEPs that shaped asyncio

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.5 Things this book did not cover, that you may want next

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Glossary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# About the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# From the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Preface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is About

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is Not

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Who This Book Is For

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Use This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Conventions Used in This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Companion Code and Data

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Contact Us

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Acknowledgments

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Contents

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Blocking Function

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.1 The first version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.2 What “blocking” actually means

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.3 The cost grows with the list

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.4 The same shape, one layer down

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.5 The question that drives the rest of the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Concurrency, Parallelism, and the GIL

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.1 What a thread is, briefly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.2 The threaded version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.3 Threading the TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.4 Why the speedup actually works (and the GIL story you have heard wrong)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.5 Why threads are not the answer the book is reaching for

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.6 The two scales the book cares about

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.7 Where the next chapter goes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Loop, From Scratch

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.1 A queue of jobs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.2 A loop that runs the next ready job

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.3 Two senses of “ready”, and the generator that gives us pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.4 The loop sleeps when nothing is ready

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.5 Connecting the toy loop to news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **3.5.1 Common mistakes you will make at least once**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Coroutines, Without the Magic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.1 Generators, the original coroutine

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.2 yield as a pause point, one more time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.3 The toy loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.4 What async def returns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Polite driver for `x in gen()`: (no for; the loop handles it)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.5 Coroutines are not running until the loop runs them

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.1 Two coroutines yielding back and forth

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.2 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### Optional: how the bridge works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async and await

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.1 What async does to a function definition

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.2 What await does at a call site

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.3 What can be awaited, and what cannot

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.4 The first real asyncio news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 5.4.1 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.5 The cliffhanger

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.6 The same keywords, on raw TCP

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# await Yields, Suspends, and Resumes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.1 What await does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.2 What is saved across the pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.3 How the loop wakes the coroutine back up

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.4 A Task is a coroutine that the loop is driving

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.5 *For the curious: coro.send(value) and the bytecode*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **6.6 *For the curious:* selectors and the operating system**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **6.7 Watching news.py yield and resume**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# asyncio.run, create\_task, gather

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.1 asyncio.run as the starting line

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.2 What asyncio.run does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.3 await coro is sequential; create\_task(coro) is concurrent

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.4 asyncio.gather for “run these together”

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.5 The fast news .py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.6 Concurrent TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.7 The lost-task trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Time Toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.1 `asyncio.sleep` yields; `time.sleep` blocks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.2 `wait_for` for a deadline around an awaitable

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.3 `asyncio.timeout()` as a context manager (Python 3.11+)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.4 `asyncio.shield` to protect from cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.5 `asyncio.wait` vs `asyncio.gather`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.6 `asyncio.as_completed` for results as they arrive

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.7 Timeouts and streaming

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async for, async with, Async Iteration and Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.1 async for and the iterator protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.2 Async generators

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.2.1 Cleanup when iteration stops short

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.3 async for over aiohttp response bodies

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.3.1 Async comprehensions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.4 async with and the context-manager protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.4.1 Multiple async with and AsyncExitStack

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.5 Writing your own async context manager

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.6 The streaming news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.6.1 Two more async-generator pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.1 Task is the loop's unit of work; the coroutine is just the recipe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.2 asyncio.TaskGroup: structured concurrency by construction

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.3 Cancellation: `task.cancel()` and `CancelledError` at the next `await`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.4 The bare-except trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.5 ExceptionGroup and except\*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.6 Re-raising CancelledError after cleanup

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.7 Canceling a TCP probe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.8 Synchronization primitives

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Lock: only one at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Semaphore: only N at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Event: one coroutine signals, others wait**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **asyncio.Queue: pass items between coroutines**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.10 contextvars for task-scoped state**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.11 Producer-consumer with asyncio.Queue**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What Goes Wrong, and How to Find It

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.1 The failure mode: one synchronous call freezes the whole loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.2 The blocking patterns to learn to recognize

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.3 The single most useful tool: asyncio's debug mode

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.4 The four warnings the runtime emits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.5 `asyncio.all_tasks()` and `task.get_stack()`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.6 aiomonitor: a live REPL inside a running program

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.7 Profiling that sees the loop correctly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.8 Logging task names

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.9 pdb and async

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.10 A guided debugging session on the broken news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.11 tcpcheck.py as a diagnostic tool

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Testing async code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.1 Your first async test

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.2 The standard-library alternative

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.3 What mocks are, and why async needs a different one

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.4 Testing cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.5 *Optional*: Testing TaskGroup error propagation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.6 Event loop fixture scope

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.7 Testing timeouts without real time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.8 The most subtle async test bug

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.9 When a test hangs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.10 The full test suite for the fetcher

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Where asyncio Ends: Executors, Graceful Shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.1 Fixing the parser bug from Chapter 11

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.2 The executor toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.3 The decision tree

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.4 Bridging from sync to async, and back

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.5 Library design: expose async, do not hide it

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.6 Graceful shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.7 The final news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Closing reflection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Chapter recap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix A: Setting up your environment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.1 Python versions and what they buy you

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.2 The packages the book uses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.3 The smallest runnable asyncio script

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.4 Reading the running example as you go

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix B: The asyncio API surface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Functions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Classes and protocols

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except*` (3.11+). 10**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Loop methods worth knowing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **loop.set\_default\_executor(executor)** Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## aihttp API surface used in the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **response.text()** Awaitable that returns the full response body as a decoded string. 5

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Where to find the rest

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix C: Troubleshooting guide

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.1 Asyncio runtime warnings, what they mean and what to fix

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeWarning: async generator 'X' was never closed** An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between `yields` in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.2 The script runs without errors but is no faster than synchronous

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **C.3 The program hangs or never exits**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.4 Common errors and what they mean

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### When the table does not have your symptom

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix D: Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.1 The asyncio source code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.2 The async library ecosystem

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.3 The structured concurrency literature

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.4 The PEPs that shaped asyncio

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.5 Things this book did not cover, that you may want next

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Glossary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# About the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# From the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Preface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is About

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is Not

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Who This Book Is For

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Use This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Conventions Used in This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Companion Code and Data

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Contact Us

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Acknowledgments

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Contents

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Blocking Function

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.1 The first version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.2 What “blocking” actually means

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.3 The cost grows with the list

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.4 The same shape, one layer down

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.5 The question that drives the rest of the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Concurrency, Parallelism, and the GIL

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.1 What a thread is, briefly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.2 The threaded version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.3 Threading the TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.4 Why the speedup actually works (and the GIL story you have heard wrong)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.5 Why threads are not the answer the book is reaching for

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.6 The two scales the book cares about

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.7 Where the next chapter goes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Loop, From Scratch

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.1 A queue of jobs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.2 A loop that runs the next ready job

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.3 Two senses of “ready”, and the generator that gives us pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.4 The loop sleeps when nothing is ready

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.5 Connecting the toy loop to news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **3.5.1 Common mistakes you will make at least once**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Coroutines, Without the Magic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.1 Generators, the original coroutine

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.2 yield as a pause point, one more time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.3 The toy loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.4 What async def returns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Polite driver for `x in gen()`: (no for; the loop handles it)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.5 Coroutines are not running until the loop runs them

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.1 Two coroutines yielding back and forth

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.2 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### Optional: how the bridge works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async and await

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.1 What async does to a function definition

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.2 What await does at a call site

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.3 What can be awaited, and what cannot

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.4 The first real asyncio news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 5.4.1 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.5 The cliffhanger

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.6 The same keywords, on raw TCP

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# await Yields, Suspends, and Resumes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.1 What await does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.2 What is saved across the pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.3 How the loop wakes the coroutine back up

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.4 A Task is a coroutine that the loop is driving

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.5 *For the curious: coro.send(value) and the bytecode*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.6 *For the curious*: selectors and the operating system

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.7 Watching news.py yield and resume

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# asyncio.run, create\_task, gather

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.1 asyncio.run as the starting line

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.2 What asyncio.run does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.3 await coro is sequential; create\_task(coro) is concurrent

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.4 asyncio.gather for “run these together”

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.5 The fast news .py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.6 Concurrent TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.7 The lost-task trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Time Toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.1 `asyncio.sleep` yields; `time.sleep` blocks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.2 `wait_for` for a deadline around an awaitable

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.3 `asyncio.timeout()` as a context manager (Python 3.11+)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.4 `asyncio.shield` to protect from cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.5 `asyncio.wait` vs `asyncio.gather`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.6 `asyncio.as_completed` for results as they arrive

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.7 Timeouts and streaming

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async for, async with, Async Iteration and Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.1 async for and the iterator protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.2 Async generators

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.2.1 Cleanup when iteration stops short

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.3 async for over aiohttp response bodies

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.3.1 Async comprehensions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.4 async with and the context-manager protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.4.1 Multiple async with and AsyncExitStack

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.5 Writing your own async context manager

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.6 The streaming news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.6.1 Two more async-generator pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.1 Task is the loop's unit of work; the coroutine is just the recipe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.2 asyncio.TaskGroup: structured concurrency by construction

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.3 Cancellation: `task.cancel()` and `CancelledError` at the next `await`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.4 The bare-except trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.5 ExceptionGroup and except\*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.6 Re-raising CancelledError after cleanup

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.7 Canceling a TCP probe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.8 Synchronization primitives

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Lock: only one at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Semaphore: only N at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Event: one coroutine signals, others wait**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **asyncio.Queue: pass items between coroutines**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.10 contextvars for task-scoped state**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.11 Producer-consumer with asyncio.Queue**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What Goes Wrong, and How to Find It

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.1 The failure mode: one synchronous call freezes the whole loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.2 The blocking patterns to learn to recognize

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.3 The single most useful tool: asyncio's debug mode

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.4 The four warnings the runtime emits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.5 `asyncio.all_tasks()` and `task.get_stack()`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.6 aiomonitor: a live REPL inside a running program

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.7 Profiling that sees the loop correctly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.8 Logging task names

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.9 pdb and async

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.10 A guided debugging session on the broken news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.11 tcpcheck.py as a diagnostic tool

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Testing async code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.1 Your first async test

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.2 The standard-library alternative

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.3 What mocks are, and why async needs a different one

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.4 Testing cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.5 *Optional*: Testing TaskGroup error propagation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.6 Event loop fixture scope

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.7 Testing timeouts without real time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.8 The most subtle async test bug

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.9 When a test hangs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.10 The full test suite for the fetcher

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Where asyncio Ends: Executors, Graceful Shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.1 Fixing the parser bug from Chapter 11

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.2 The executor toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.3 The decision tree

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.4 Bridging from sync to async, and back

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.5 Library design: expose async, do not hide it

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.6 Graceful shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.7 The final news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Closing reflection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Chapter recap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix A: Setting up your environment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.1 Python versions and what they buy you

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.2 The packages the book uses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.3 The smallest runnable asyncio script

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.4 Reading the running example as you go

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix B: The asyncio API surface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Functions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Classes and protocols

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except*` (3.11+). 10**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Loop methods worth knowing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **loop.set\_default\_executor(executor)** Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## aihttp API surface used in the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **response.text()** Awaitable that returns the full response body as a decoded string. 5

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Where to find the rest

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix C: Troubleshooting guide

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.1 Asyncio runtime warnings, what they mean and what to fix

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeWarning: async generator 'X' was never closed** An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between `yields` in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.2 The script runs without errors but is no faster than synchronous

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **C.3 The program hangs or never exits**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.4 Common errors and what they mean

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### When the table does not have your symptom

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix D: Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.1 The asyncio source code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.2 The async library ecosystem

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.3 The structured concurrency literature

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.4 The PEPs that shaped asyncio

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.5 Things this book did not cover, that you may want next

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Glossary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# About the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# From the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Preface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is About

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is Not

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Who This Book Is For

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Use This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Conventions Used in This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Companion Code and Data

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Contact Us

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Acknowledgments

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Contents

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Blocking Function

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.1 The first version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.2 What “blocking” actually means

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.3 The cost grows with the list

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.4 The same shape, one layer down

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.5 The question that drives the rest of the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Concurrency, Parallelism, and the GIL

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.1 What a thread is, briefly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.2 The threaded version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.3 Threading the TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.4 Why the speedup actually works (and the GIL story you have heard wrong)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.5 Why threads are not the answer the book is reaching for

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.6 The two scales the book cares about

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.7 Where the next chapter goes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Loop, From Scratch

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.1 A queue of jobs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.2 A loop that runs the next ready job

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.3 Two senses of “ready”, and the generator that gives us pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.4 The loop sleeps when nothing is ready

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.5 Connecting the toy loop to news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **3.5.1 Common mistakes you will make at least once**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Coroutines, Without the Magic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.1 Generators, the original coroutine

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.2 yield as a pause point, one more time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.3 The toy loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.4 What async def returns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Polite driver for `x in gen()`: (no for; the loop handles it)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.5 Coroutines are not running until the loop runs them

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.1 Two coroutines yielding back and forth

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.2 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### Optional: how the bridge works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async and await

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.1 What async does to a function definition

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.2 What await does at a call site

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.3 What can be awaited, and what cannot

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.4 The first real asyncio news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 5.4.1 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.5 The cliffhanger

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.6 The same keywords, on raw TCP

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# await Yields, Suspends, and Resumes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.1 What await does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.2 What is saved across the pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.3 How the loop wakes the coroutine back up

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.4 A Task is a coroutine that the loop is driving

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.5 *For the curious: coro.send(value) and the bytecode*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.6 *For the curious*: selectors and the operating system

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.7 Watching news.py yield and resume

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# asyncio.run, create\_task, gather

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.1 asyncio.run as the starting line

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.2 What asyncio.run does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.3 await coro is sequential; create\_task(coro) is concurrent

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.4 asyncio.gather for “run these together”

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.5 The fast news .py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.6 Concurrent TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.7 The lost-task trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Time Toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.1 `asyncio.sleep` yields; `time.sleep` blocks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.2 `wait_for` for a deadline around an awaitable

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.3 `asyncio.timeout()` as a context manager (Python 3.11+)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.4 `asyncio.shield` to protect from cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.5 `asyncio.wait` vs `asyncio.gather`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.6 `asyncio.as_completed` for results as they arrive

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.7 Timeouts and streaming

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async for, async with, Async Iteration and Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.1 async for and the iterator protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.2 Async generators

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.2.1 Cleanup when iteration stops short

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.3 async for over aiohttp response bodies

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.3.1 Async comprehensions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.4 async with and the context-manager protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.4.1 Multiple async with and AsyncExitStack

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.5 Writing your own async context manager

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.6 The streaming news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.6.1 Two more async-generator pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.1 Task is the loop's unit of work; the coroutine is just the recipe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.2 asyncio.TaskGroup: structured concurrency by construction

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.3 Cancellation: `task.cancel()` and `CancelledError` at the next `await`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.4 The bare-except trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.5 ExceptionGroup and except\*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.6 Re-raising CancelledError after cleanup

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.7 Canceling a TCP probe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.8 Synchronization primitives

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Lock: only one at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Semaphore: only N at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Event: one coroutine signals, others wait**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **asyncio.Queue: pass items between coroutines**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.10 contextvars for task-scoped state**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.11 Producer-consumer with asyncio.Queue**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What Goes Wrong, and How to Find It

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.1 The failure mode: one synchronous call freezes the whole loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.2 The blocking patterns to learn to recognize

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.3 The single most useful tool: asyncio's debug mode

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.4 The four warnings the runtime emits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.5 `asyncio.all_tasks()` and `task.get_stack()`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.6 aiomonitor: a live REPL inside a running program

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.7 Profiling that sees the loop correctly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.8 Logging task names

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.9 pdb and async

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.10 A guided debugging session on the broken news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.11 tcpcheck.py as a diagnostic tool

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Testing async code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.1 Your first async test

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.2 The standard-library alternative

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.3 What mocks are, and why async needs a different one

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.4 Testing cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.5 *Optional*: Testing TaskGroup error propagation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.6 Event loop fixture scope

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.7 Testing timeouts without real time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.8 The most subtle async test bug

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.9 When a test hangs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.10 The full test suite for the fetcher

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Where asyncio Ends: Executors, Graceful Shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.1 Fixing the parser bug from Chapter 11

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.2 The executor toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.3 The decision tree

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.4 Bridging from sync to async, and back

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.5 Library design: expose async, do not hide it

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.6 Graceful shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.7 The final news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Closing reflection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Chapter recap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix A: Setting up your environment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.1 Python versions and what they buy you

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.2 The packages the book uses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.3 The smallest runnable asyncio script

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.4 Reading the running example as you go

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix B: The asyncio API surface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Functions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Classes and protocols

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except*` (3.11+). 10**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Loop methods worth knowing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **loop.set\_default\_executor(executor)** Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## aihttp API surface used in the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **response.text()** Awaitable that returns the full response body as a decoded string. 5

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Where to find the rest

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix C: Troubleshooting guide

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.1 Asyncio runtime warnings, what they mean and what to fix

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeWarning: async generator 'X' was never closed** An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between `yields` in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.2 The script runs without errors but is no faster than synchronous

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **C.3 The program hangs or never exits**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.4 Common errors and what they mean

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### When the table does not have your symptom

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix D: Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.1 The asyncio source code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.2 The async library ecosystem

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.3 The structured concurrency literature

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.4 The PEPs that shaped asyncio

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.5 Things this book did not cover, that you may want next

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Glossary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# About the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# From the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Preface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is About

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is Not

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Who This Book Is For

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Use This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Conventions Used in This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Companion Code and Data

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Contact Us

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Acknowledgments

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Contents

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Blocking Function

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.1 The first version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.2 What “blocking” actually means

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.3 The cost grows with the list

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.4 The same shape, one layer down

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.5 The question that drives the rest of the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Concurrency, Parallelism, and the GIL

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.1 What a thread is, briefly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.2 The threaded version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.3 Threading the TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.4 Why the speedup actually works (and the GIL story you have heard wrong)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.5 Why threads are not the answer the book is reaching for

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.6 The two scales the book cares about

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.7 Where the next chapter goes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Loop, From Scratch

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.1 A queue of jobs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.2 A loop that runs the next ready job

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.3 Two senses of “ready”, and the generator that gives us pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.4 The loop sleeps when nothing is ready

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.5 Connecting the toy loop to news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **3.5.1 Common mistakes you will make at least once**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Coroutines, Without the Magic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.1 Generators, the original coroutine

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.2 yield as a pause point, one more time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.3 The toy loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.4 What async def returns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Polite driver for `x in gen()`: (no for; the loop handles it)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.5 Coroutines are not running until the loop runs them

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.1 Two coroutines yielding back and forth

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.2 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### Optional: how the bridge works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async and await

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.1 What async does to a function definition

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.2 What await does at a call site

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.3 What can be awaited, and what cannot

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.4 The first real asyncio news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 5.4.1 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.5 The cliffhanger

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.6 The same keywords, on raw TCP

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# await Yields, Suspends, and Resumes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.1 What await does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.2 What is saved across the pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.3 How the loop wakes the coroutine back up

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.4 A Task is a coroutine that the loop is driving

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.5 *For the curious: coro.send(value) and the bytecode*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.6 *For the curious*: selectors and the operating system

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.7 Watching news.py yield and resume

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# asyncio.run, create\_task, gather

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.1 asyncio.run as the starting line

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.2 What asyncio.run does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.3 await coro is sequential; create\_task(coro) is concurrent

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.4 asyncio.gather for “run these together”

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.5 The fast news .py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.6 Concurrent TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.7 The lost-task trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Time Toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.1 `asyncio.sleep` yields; `time.sleep` blocks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.2 `wait_for` for a deadline around an awaitable

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.3 `asyncio.timeout()` as a context manager (Python 3.11+)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.4 `asyncio.shield` to protect from cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.5 `asyncio.wait` vs `asyncio.gather`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.6 `asyncio.as_completed` for results as they arrive

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.7 Timeouts and streaming

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async for, async with, Async Iteration and Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.1 async for and the iterator protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.2 Async generators

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.2.1 Cleanup when iteration stops short

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.3 async for over aiohttp response bodies

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.3.1 Async comprehensions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.4 async with and the context-manager protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.4.1 Multiple async with and AsyncExitStack

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.5 Writing your own async context manager

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.6 The streaming news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.6.1 Two more async-generator pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.1 Task is the loop's unit of work; the coroutine is just the recipe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.2 asyncio.TaskGroup: structured concurrency by construction

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.3 Cancellation: `task.cancel()` and `CancelledError` at the next `await`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.4 The bare-except trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.5 ExceptionGroup and except\*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.6 Re-raising CancelledError after cleanup

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.7 Canceling a TCP probe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.8 Synchronization primitives

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Lock: only one at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Semaphore: only N at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Event: one coroutine signals, others wait**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **asyncio.Queue: pass items between coroutines**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.10 contextvars for task-scoped state**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.11 Producer-consumer with asyncio.Queue**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What Goes Wrong, and How to Find It

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.1 The failure mode: one synchronous call freezes the whole loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.2 The blocking patterns to learn to recognize

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.3 The single most useful tool: asyncio's debug mode

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.4 The four warnings the runtime emits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.5 `asyncio.all_tasks()` and `task.get_stack()`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.6 aiomonitor: a live REPL inside a running program

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.7 Profiling that sees the loop correctly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.8 Logging task names

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.9 pdb and async

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.10 A guided debugging session on the broken news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.11 tcpcheck.py as a diagnostic tool

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Testing async code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.1 Your first async test

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.2 The standard-library alternative

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.3 What mocks are, and why async needs a different one

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.4 Testing cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.5 *Optional*: Testing TaskGroup error propagation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.6 Event loop fixture scope

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.7 Testing timeouts without real time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.8 The most subtle async test bug

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.9 When a test hangs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.10 The full test suite for the fetcher

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Where asyncio Ends: Executors, Graceful Shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.1 Fixing the parser bug from Chapter 11

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.2 The executor toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.3 The decision tree

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.4 Bridging from sync to async, and back

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.5 Library design: expose async, do not hide it

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.6 Graceful shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.7 The final news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Closing reflection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Chapter recap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix A: Setting up your environment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.1 Python versions and what they buy you

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.2 The packages the book uses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.3 The smallest runnable asyncio script

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.4 Reading the running example as you go

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix B: The asyncio API surface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Functions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Classes and protocols

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except*` (3.11+). 10**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Loop methods worth knowing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`loop.set_default_executor(executor)` Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## aihttp API surface used in the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`response.text()` Awaitable that returns the full response body as a decoded string. 5**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Where to find the rest

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix C: Troubleshooting guide

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.1 Asyncio runtime warnings, what they mean and what to fix

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeWarning: async generator 'X' was never closed** An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between `yields` in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.2 The script runs without errors but is no faster than synchronous

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **C.3 The program hangs or never exits**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.4 Common errors and what they mean

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### When the table does not have your symptom

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix D: Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.1 The asyncio source code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.2 The async library ecosystem

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.3 The structured concurrency literature

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.4 The PEPs that shaped asyncio

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.5 Things this book did not cover, that you may want next

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Glossary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# About the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# From the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Preface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is About

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is Not

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Who This Book Is For

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Use This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Conventions Used in This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Companion Code and Data

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Contact Us

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Acknowledgments

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Contents

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Blocking Function

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.1 The first version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.2 What “blocking” actually means

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.3 The cost grows with the list

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.4 The same shape, one layer down

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.5 The question that drives the rest of the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Concurrency, Parallelism, and the GIL

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.1 What a thread is, briefly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.2 The threaded version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.3 Threading the TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.4 Why the speedup actually works (and the GIL story you have heard wrong)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.5 Why threads are not the answer the book is reaching for

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.6 The two scales the book cares about

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.7 Where the next chapter goes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Loop, From Scratch

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.1 A queue of jobs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.2 A loop that runs the next ready job

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.3 Two senses of “ready”, and the generator that gives us pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.4 The loop sleeps when nothing is ready

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.5 Connecting the toy loop to news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **3.5.1 Common mistakes you will make at least once**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Coroutines, Without the Magic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.1 Generators, the original coroutine

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.2 yield as a pause point, one more time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.3 The toy loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.4 What async def returns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Polite driver for `x in gen()`: (no for; the loop handles it)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.5 Coroutines are not running until the loop runs them

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.1 Two coroutines yielding back and forth

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.2 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### Optional: how the bridge works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async and await

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.1 What async does to a function definition

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.2 What await does at a call site

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.3 What can be awaited, and what cannot

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.4 The first real asyncio news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 5.4.1 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.5 The cliffhanger

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.6 The same keywords, on raw TCP

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# await Yields, Suspends, and Resumes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.1 What await does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.2 What is saved across the pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.3 How the loop wakes the coroutine back up

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.4 A Task is a coroutine that the loop is driving

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.5 *For the curious: coro.send(value) and the bytecode*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.6 *For the curious*: selectors and the operating system

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.7 Watching news.py yield and resume

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# asyncio.run, create\_task, gather

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.1 asyncio.run as the starting line

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.2 What asyncio.run does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.3 await coro is sequential; create\_task(coro) is concurrent

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.4 asyncio.gather for “run these together”

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.5 The fast news .py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.6 Concurrent TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.7 The lost-task trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Time Toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.1 `asyncio.sleep` yields; `time.sleep` blocks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.2 `wait_for` for a deadline around an awaitable

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.3 `asyncio.timeout()` as a context manager (Python 3.11+)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.4 `asyncio.shield` to protect from cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.5 `asyncio.wait` vs `asyncio.gather`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.6 `asyncio.as_completed` for results as they arrive

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.7 Timeouts and streaming

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async for, async with, Async Iteration and Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.1 async for and the iterator protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.2 Async generators

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.2.1 Cleanup when iteration stops short

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.3 async for over aiohttp response bodies

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.3.1 Async comprehensions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.4 async with and the context-manager protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.4.1 Multiple async with and AsyncExitStack

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.5 Writing your own async context manager

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.6 The streaming news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.6.1 Two more async-generator pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.1 Task is the loop's unit of work; the coroutine is just the recipe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.2 asyncio.TaskGroup: structured concurrency by construction

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.3 Cancellation: `task.cancel()` and `CancelledError` at the next `await`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.4 The bare-except trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.5 ExceptionGroup and except\*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.6 Re-raising CancelledError after cleanup

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.7 Canceling a TCP probe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.8 Synchronization primitives

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Lock: only one at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Semaphore: only N at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Event: one coroutine signals, others wait**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **asyncio.Queue: pass items between coroutines**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.10 contextvars for task-scoped state**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.11 Producer-consumer with asyncio.Queue**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What Goes Wrong, and How to Find It

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.1 The failure mode: one synchronous call freezes the whole loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.2 The blocking patterns to learn to recognize

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.3 The single most useful tool: asyncio's debug mode

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.4 The four warnings the runtime emits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.5 `asyncio.all_tasks()` and `task.get_stack()`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.6 aiomonitor: a live REPL inside a running program

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.7 Profiling that sees the loop correctly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.8 Logging task names

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.9 pdb and async

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.10 A guided debugging session on the broken news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.11 tcpcheck.py as a diagnostic tool

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Testing async code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.1 Your first async test

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.2 The standard-library alternative

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.3 What mocks are, and why async needs a different one

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.4 Testing cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.5 *Optional*: Testing TaskGroup error propagation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.6 Event loop fixture scope

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.7 Testing timeouts without real time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.8 The most subtle async test bug

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.9 When a test hangs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.10 The full test suite for the fetcher

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Where asyncio Ends: Executors, Graceful Shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.1 Fixing the parser bug from Chapter 11

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.2 The executor toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.3 The decision tree

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.4 Bridging from sync to async, and back

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.5 Library design: expose async, do not hide it

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.6 Graceful shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.7 The final news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Closing reflection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Chapter recap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix A: Setting up your environment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.1 Python versions and what they buy you

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.2 The packages the book uses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.3 The smallest runnable asyncio script

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.4 Reading the running example as you go

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix B: The asyncio API surface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Functions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Classes and protocols

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except*` (3.11+). 10**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Loop methods worth knowing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **loop.set\_default\_executor(executor)** Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## aihttp API surface used in the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **response.text()** Awaitable that returns the full response body as a decoded string. 5

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Where to find the rest

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix C: Troubleshooting guide

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.1 Asyncio runtime warnings, what they mean and what to fix

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeWarning: async generator 'X' was never closed** An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between `yields` in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.2 The script runs without errors but is no faster than synchronous

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **C.3 The program hangs or never exits**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.4 Common errors and what they mean

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### When the table does not have your symptom

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix D: Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.1 The asyncio source code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.2 The async library ecosystem

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.3 The structured concurrency literature

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.4 The PEPs that shaped asyncio

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.5 Things this book did not cover, that you may want next

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Glossary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# About the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# From the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Preface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is About

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What This Book Is Not

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Who This Book Is For

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Use This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Conventions Used in This Book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Companion Code and Data

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# How to Contact Us

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Acknowledgments

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Contents

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Blocking Function

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.1 The first version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.2 What “blocking” actually means

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.3 The cost grows with the list

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.4 The same shape, one layer down

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 1.5 The question that drives the rest of the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Concurrency, Parallelism, and the GIL

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.1 What a thread is, briefly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.2 The threaded version of news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.3 Threading the TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.4 Why the speedup actually works (and the GIL story you have heard wrong)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.5 Why threads are not the answer the book is reaching for

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.6 The two scales the book cares about

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 2.7 Where the next chapter goes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Loop, From Scratch

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.1 A queue of jobs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.2 A loop that runs the next ready job

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.3 Two senses of “ready”, and the generator that gives us pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.4 The loop sleeps when nothing is ready

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 3.5 Connecting the toy loop to news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **3.5.1 Common mistakes you will make at least once**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Coroutines, Without the Magic

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.1 Generators, the original coroutine

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.2 yield as a pause point, one more time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.3 The toy loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.4 What async def returns

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Polite driver for `x in gen()`: (no for; the loop handles it)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 4.5 Coroutines are not running until the loop runs them

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.1 Two coroutines yielding back and forth

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 4.5.2 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### Optional: how the bridge works

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async and await

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.1 What async does to a function definition

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.2 What await does at a call site

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.3 What can be awaited, and what cannot

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.4 The first real asyncio news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 5.4.1 Common mistakes you will make at least once

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.5 The cliffhanger

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 5.6 The same keywords, on raw TCP

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# await Yields, Suspends, and Resumes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.1 What await does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.2 What is saved across the pause

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.3 How the loop wakes the coroutine back up

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.4 A Task is a coroutine that the loop is driving

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.5 *For the curious: coro.send(value) and the bytecode*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.6 *For the curious*: selectors and the operating system

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 6.7 Watching news.py yield and resume

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# asyncio.run, create\_task, gather

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.1 asyncio.run as the starting line

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.2 What asyncio.run does

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.3 await coro is sequential; create\_task(coro) is concurrent

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.4 asyncio.gather for “run these together”

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.5 The fast news .py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.6 Concurrent TCP probes

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 7.7 The lost-task trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# The Time Toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.1 `asyncio.sleep` yields; `time.sleep` blocks

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.2 `wait_for` for a deadline around an awaitable

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.3 `asyncio.timeout()` as a context manager (Python 3.11+)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.4 `asyncio.shield` to protect from cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.5 `asyncio.wait` vs `asyncio.gather`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.6 `asyncio.as_completed` for results as they arrive

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 8.7 Timeouts and streaming

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# async for, async with, Async Iteration and Resource Management

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.1 async for and the iterator protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.2 Async generators

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.2.1 Cleanup when iteration stops short

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.3 async for over aiohttp response bodies

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.3.1 Async comprehensions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.4 async with and the context-manager protocol

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.4.1 Multiple async with and AsyncExitStack

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.5 Writing your own async context manager

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 9.6 The streaming news . py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### 9.6.1 Two more async-generator pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Tasks, TaskGroups, Cancellation, Exception Groups, and contextvars

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.1 Task is the loop's unit of work; the coroutine is just the recipe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.2 asyncio.TaskGroup: structured concurrency by construction

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.3 Cancellation: `task.cancel()` and `CancelledError` at the next `await`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.4 The bare-except trap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.5 ExceptionGroup and except\*

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.6 Re-raising CancelledError after cleanup

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.7 Canceling a TCP probe

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 10.8 Synchronization primitives

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Lock: only one at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Semaphore: only N at a time**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **asyncio.Event: one coroutine signals, others wait**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **asyncio.Queue: pass items between coroutines**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.9 Capping concurrency with Semaphore: 50 sites, 10 in flight**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.10 contextvars for task-scoped state**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **10.11 Producer-consumer with asyncio.Queue**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## **Summary**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# What Goes Wrong, and How to Find It

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.1 The failure mode: one synchronous call freezes the whole loop

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.2 The blocking patterns to learn to recognize

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.3 The single most useful tool: asyncio's debug mode

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.4 The four warnings the runtime emits

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.5 `asyncio.all_tasks()` and `task.get_stack()`

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.6 aiomonitor: a live REPL inside a running program

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.7 Profiling that sees the loop correctly

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.8 Logging task names

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.9 pdb and async

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.10 A guided debugging session on the broken news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 11.11 tcpcheck.py as a diagnostic tool

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Testing async code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.1 Your first async test

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.2 The standard-library alternative

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.3 What mocks are, and why async needs a different one

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.4 Testing cancellation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.5 *Optional*: Testing TaskGroup error propagation

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.6 Event loop fixture scope

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.7 Testing timeouts without real time

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.8 The most subtle async test bug

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.9 When a test hangs

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 12.10 The full test suite for the fetcher

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Where asyncio Ends: Executors, Graceful Shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.1 Fixing the parser bug from Chapter 11

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.2 The executor toolbox

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.3 The decision tree

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Heavy CPU work that needs to scale across cores Use multiprocessing directly. asyncio is the wrong tool for that workload.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.4 Bridging from sync to async, and back

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.5 Library design: expose async, do not hide it

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.6 Graceful shutdown

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## 13.7 The final news.py

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Summary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Closing reflection

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Chapter recap

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix A: Setting up your environment

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.1 Python versions and what they buy you

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.2 The packages the book uses

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.3 The smallest runnable asyncio script

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## A.4 Reading the running example as you go

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix B: The asyncio API surface

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Functions

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`asyncio.wait_for(coro, timeout)` Wraps a single awaitable with a deadline; cancels the inner work and raises `TimeoutError` on expiry. 8**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Classes and protocols

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`ExceptionGroup` The exception type that holds multiple exceptions raised by sibling `Tasks` in a `TaskGroup`. Caught with `except*` (3.11+). 10**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Loop methods worth knowing

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`loop.set_default_executor(executor)` Replaces the loop's default executor. Useful when the default thread pool is too small for the workload. 13**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## aihttp API surface used in the book

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**`response.text()` Awaitable that returns the full response body as a decoded string. 5**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## Where to find the rest

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix C: Troubleshooting guide

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.1 Asyncio runtime warnings, what they mean and what to fix

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeWarning: async generator 'X' was never closed** An async generator was garbage-collected mid-iteration without the consumer running it to completion or calling `aclose()`. Wrap resources held between `yields` in `try/finally`. Call `aclose()` explicitly when you need cleanup at a specific moment. (Chapter 9, Section 9.2.1.)

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.2 The script runs without errors but is no faster than synchronous

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Slow-callback warnings fire (Executing <Task ...> took 0.234 seconds) when running with debug=True. A coroutine ran for too long between awaits, blocking the loop. Usually a CPU-bound function or a sync I/O call. Move the slow work to a thread pool with `asyncio.to_thread`. For CPU-bound work that needs cores, use a process pool via `loop.run_in_executor(ProcessPoolExecutor(), ...)`. (Chapter 11; Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### **C.3 The program hangs or never exits**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**Ctrl+C does not interrupt the program. No signal handler installed; or a signal handler that schedules async work but `asyncio.run` is blocked elsewhere. Install signal handlers with `loop.add_signal_handler(sig, callback)`. Cancel the top-level Task in the callback. The TaskGroup propagates cancellation to children. (Chapter 13.)**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## C.4 Common errors and what they mean

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

**RuntimeError: Event loop is closed (often during program shutdown) Code is trying to use the loop after `asyncio.run` returned and closed it. Common when a finalizer or `__del__` method tries to schedule async work. Move any async cleanup into the running coroutine's normal exit path (`finally` blocks, `__aexit__`). Do not rely on `__del__` or `atexit` for async cleanup.**

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

### When the table does not have your symptom

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Appendix D: Further reading

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.1 The asyncio source code

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.2 The async library ecosystem

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.3 The structured concurrency literature

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.4 The PEPs that shaped asyncio

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

## D.5 Things this book did not cover, that you may want next

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# Glossary

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.

# About the Author

This content is not available in the sample book. The book can be purchased on Leanpub at <https://leanpub.com/asyncio>.