

CONTENTS IN DETAIL

FOREWORD	xxi
-----------------	------------

INTRODUCTION	xxiii
---------------------	--------------

The Birth of TypeScriptxxiii
Why Now?xxiv
Who Is This Book For?xxiv
What We'll Coverxxv
Online Resourcesxxvi
The Book's Exercisesxxvi
@ts-expect-errorxxvii
Vitestxxvii

PART I: GETTING STARTED	1
--------------------------------	----------

1	
KICKSTART YOUR TYPESCRIPT SETUP	3

What's Different About TypeScript?	3
A High-Level View of How TypeScript Works	4
Tools for TypeScript Development	5
Installing Node.js	6
Installing the pnpm Package Manager	6
Installing TypeScript	7
Summary	7

2	
IDE SUPERPOWERS	9

Autocomplete	9
Manual Autocomplete	10
Exercise 2-1: Autocomplete	11
TypeScript Error Checking	11
Runtime Errors	11
Non-Runtime Errors	12
Warning Locations	12
Multiline Errors	13
Introspecting Variables and Declarations	14
Exercise 2-2: Hovering Over a Function Call	15
JSDoc Comments	16
Exercise 2-3: Adding Documentation for Hovers	16
Navigating with Go to Definition and Go to References	17
Rename Symbol	18
Automatic Imports	18
Quick Fixes	19

Restarting the VS Code Server	19
Working in JavaScript	19
Exercise 2-4: Quick Fix Refactoring	20
Summary	22

3

TYPESCRIPT IN THE DEVELOPMENT PIPELINE **23**

The Problem with TypeScript in the Browser	24
Transpiling TypeScript	25
Initializing a TypeScript Project	25
Running tsc	26
Does TypeScript Change Your JavaScript?	26
A Note on Version Control	27
Running TypeScript in Watch Mode	27
Errors in the TypeScript CLI	27
TypeScript with Modern Frameworks	28
TypeScript as a Linter	28
Summary	28

PART II: FUNDAMENTALS **31**

4

ESSENTIAL TYPES AND ANNOTATIONS **33**

Type Annotations	33
The Basic Types	34
Function Parameter Annotations	34
Variable Annotations	35
Type Inference	35
The any Type	37
Exercise 4-1: Basic Types with Function Parameters	38
Exercise 4-2: Annotating Empty Parameters	39
Exercise 4-3: The Basic Types	40
Exercise 4-4: The any Type	41
Object Literal Types	43
Optional Object Properties	43
Exercise 4-5: Object Literal Types	43
Exercise 4-6: Optional Property Types	44
Type Aliases	46
Sharing Types Across Modules	46
Exercise 4-7: The type Keyword	47
Arrays	48
Arrays of Objects	49
Tuples	50
Named Tuples	50
Exercise 4-8: Array Type	51
Exercise 4-9: Arrays of Objects	52
Exercise 4-10: Tuples	54
Exercise 4-11: Optional Members of Tuples	55

Passing Types to Functions	56
Passing Types to Set	56
Not All Functions Can Receive Types	57
Exercise 4-12: Passing Types to Map	58
Exercise 4-13: JSON.parse() Can't Receive Type Arguments	59
Typing Functions	61
Optional Parameters	61
Default Parameters	61
Function Return Types	62
Rest Parameters.	62
Function Types	63
The void Type	64
Async Functions	64
Exercise 4-14: Optional Function Parameters	65
Exercise 4-15: Default Function Parameters	66
Exercise 4-16: Rest Parameters	67
Exercise 4-17: Function Types	68
Exercise 4-18: Functions Returning void	71
Exercise 4-19: void vs. undefined	72
Exercise 4-20: Async Functions	73
Summary	75

5 UNIONS, LITERALS, AND NARROWING **77**

Union Types	77
Declaring Union Types.	78
Literal Types.	79
Combining Unions with Unions.	79
Exercise 5-1: string or null.	80
Exercise 5-2: Restricting Function Parameters.	81
Wide and Narrow Types	82
Unions Are Wider Than Their Members	83
The Process of Narrowing	84
Exercise 5-3: Narrowing with if Statements	85
Exercise 5-4: Throwing Errors to Narrow	87
Exercise 5-5: Using in to Narrow	88
The unknown and never Types	90
The Widest Type: unknown.	90
The Difference Between unknown and any	91
The Narrowest Type: never.	92
Exercise 5-6: Narrowing Errors with instanceof	93
Exercise 5-7: Narrowing unknown to a Value	95
Discriminated Unions	97
The Problem: The Bag of Optionals	97
The Solution: Discriminated Unions	98
Exercise 5-8: Destructuring a Discriminated Union	100
Exercise 5-9: Narrowing a Discriminated Union with a switch Statement.	102
Exercise 5-10: Discriminated Tuples	103
Exercise 5-11: Handling Defaults with a Discriminated Union	105
Summary	108

PART III: OBJECTS, CLASSES, AND MUTABILITY

109

6		
OBJECTS		111
Extending Objects		111
Intersection Types		112
Interfaces		113
Intersections vs. interface extends		115
Types vs. Interfaces		116
Exercise 6-1: Creating an Intersection Type		117
Exercise 6-2: Extending Interfaces		119
Dynamic Object Keys		120
Index Signatures		121
The Record Type		122
A Combination of Known and Dynamic Keys		122
The PropertyKey Type		123
Exercise 6-3: Using an Index Signature for Dynamic Keys		124
Exercise 6-4: Default Properties with Dynamic Keys		125
Exercise 6-5: Restricting Object Keys with Records		126
Exercise 6-6: Dynamic Key Support		127
Reducing Duplication with Utility Types		129
The Partial Type		129
The Required Type		129
The Pick Type		131
The Omit Type		131
Union Types with Omit and Pick		132
Exercise 6-7: Expecting Certain Properties		135
Exercise 6-8: Updating a Product		136
Summary		138
7		
MUTABILITY		139
Mutability and Inference		139
How TypeScript Infers let		140
How TypeScript Infers const		140
Object Property Inference		141
Readonly Object Properties		143
Exercise 7-1: Inference with an Array of Objects		146
Exercise 7-2: Avoiding Array Mutation		148
Exercise 7-3: An Unsafe Tuple		149
Deep Immutability with as const		150
as const vs. Variable Annotation		151
as const vs. Object.freeze		151
Exercise 7-4: Inferring Literal Values in Arrays		153
Summary		156

8

CLASSES

159

Creating a Class	159
Adding a Constructor	160
Adding Arguments to the Constructor.	161
Using a Class as a Type.	161
Properties in Classes.	162
Class Property Initializers.	162
readonly Class Properties.	163
Optional Class Properties.	163
public and private Properties	163
Class Methods.	165
Class Inheritance	166
Extending a Class	167
protected Properties.	168
Safe Overrides with override	168
The implements Keyword	169
Abstract Classes	171
Abstract Methods	172
Exercise 8-1: Creating a Class.	173
Exercise 8-2: Implementing Class Methods.	174
Exercise 8-3: Implementing a Getter	175
Exercise 8-4: Implementing a Setter.	177
Exercise 8-5: Extending a Class.	177
Summary	179

9

TYPESCRIPT-ONLY FEATURES

181

Class Parameter Properties	181
Enums.	182
Numeric Enums.	182
String Enums.	183
Enums Are Strange	184
Should You Use Enums?.	187
Namespaces	188
How Namespaces Compile	189
Merging Namespaces.	189
Should You Use Namespaces?	191
Type-Only Namespaces	191
When to Use ECMAScript vs. TypeScript.	191
The Future of TypeScript: Erasable Syntax Only	192
Summary	192

10 DERIVING TYPES 197

- Deriving Types from Other Types 198
 - The keyof Operator 198
 - The typeof Operator 199
- You Can't Create Values from Types 200
- Indexed Access Types 201
 - Chaining Multiple Indexed Access Types 202
 - Passing a Union to an Indexed Access Type 202
 - Getting an Object's Values with keyof 203
- Using as const for JavaScript-Style Enums 203
 - Enums Require You to Pass the Enum Value 204
 - Enums Are Nominal 204
 - Which Approach Should You Use? 205
 - Exercise 10-1: Reducing Key Repetition 205**
 - Exercise 10-2: Deriving a Type from a Value 206**
 - Exercise 10-3: Accessing Specific Values 207**
 - Exercise 10-4: Unions with Indexed Access Types 208**
 - Exercise 10-5: Extract a Union of All Values 209**
 - Exercise 10-6: Creating a Union from an as const Array 210**
- Deriving Types from Functions 212
 - Parameters 212
 - ReturnType 213
 - Awaited 213
- Why Derive Types from Functions? 213
 - Exercise 10-7: A Single Source of Truth 214**
 - Exercise 10-8: Typing Based on a Return Value 216**
 - Exercise 10-9: Unwrapping a Promise 217**
- Transforming Derived Types 218
 - Exclude 218
 - NonNullable 219
 - Extract 220
 - Deriving vs. Decoupling 221
 - When Decoupling Makes Sense 221
 - When Deriving Makes Sense 222
- Summary 223

11 ANNOTATIONS AND ASSERTIONS 225

- Annotating Variables vs. Values 225
 - Annotating Values with satisfies 227
 - Narrowing Values with satisfies 228
- Assertions: Forcing the Type of Values 229
 - The as Assertion 229
 - The Non-Null Assertion 231
- Error Suppression Directives 232
 - @ts-expect-error 232
 - @ts-ignore 233

@ts-nocheck	233
Suppressing Errors vs. as any	234
When to Suppress Errors	234
When You Know More Than TypeScript	235
When TypeScript Is Being “Dumb”	235
When You Don’t Understand the Error	236
Exercise 11-1: Providing Additional Info to TypeScript	236
Exercise 11-2: Solving Issues with Assertions	238
Exercise 11-3: Enforcing a Valid Configuration	240
Exercise 11-4: Variable Annotation vs. as vs. satisfies	242
Exercise 11-5: Creating a Deeply Read-Only Object	246
Summary	248

12

THE WEIRD PARTS 249

The Evolving any Type	249
Excess Property Warnings	251
No Excess Property Checks on Variables	251
No Excess Property Checks When Comparing Functions	252
Open vs. Closed Object Types	253
Fresh and Stale Objects	254
Object Keys Are Loosely Typed	255
The Empty Object Type	256
The Type and Value Worlds	258
Classes Can Cross Between Worlds	259
Enums Can Cross Between Worlds	260
The this Keyword Can Cross Between Worlds	260
Naming Types and Values the Same	261
Using this in Functions	263
Arrow Functions Don’t Support this	265
Function Assignability	265
Unions of Functions Intersect Parameters	267
Exercise 12-1: Accepting Anything Except null and undefined	269
Exercise 12-2: Detecting Excess Properties in an Object	270
Exercise 12-3: Detecting Excess Properties in a Function	272
Exercise 12-4: Iterating over Objects	274
Exercise 12-5: Function Parameter Comparisons	276
Exercise 12-6: Unions of Functions with Object Params	278
Exercise 12-7: Unions of Functions with Incompatible Parameters	279
Summary	281

PART V: UNDERSTANDING THE ENVIRONMENT

283

13

MODULES, SCRIPTS, AND DECLARATION FILES 285

Understanding Modules and Scripts	285
Modules Have Local Scope	286
Scripts Have Global Scope	286

TypeScript Guesses Which to Use	287
Forcing Modules with moduleDetection	288
Declaration Files	288
Declaration Files Describe JavaScript.	288
Declaration Files Can Add to the Global Scope	290
Declaration Files Can't Contain Implementations	290
The declare Keyword	291
Typing Global Variables.	291
Scoping Global Variables to One File	292
More Ways to declare.	292
Module Augmentation vs. Module Overriding.	294
Declaration Files You Don't Control	295
TypeScript's Types	295
DOM Types	296
Types That Ship with Libraries.	297
DefinitelyTyped	298
The skipLibCheck Config Option.	299
Authoring Declaration Files	299
Should You Store Your Types in Declaration Files?	301
Is Using Global Types a Good Idea?	301
Exercise 13-1: Typing a JavaScript Module	302
Exercise 13-2: Ambient Context.	302
Exercise 13-3: Modifying window.	304
Exercise 13-4: Modifying process.env	305
Summary	306

14

CONFIGURING TYPESCRIPT	307
Recommended Configuration.	307
Additional Configuration Options	308
The Complete Base Configuration	309
Base Options.	310
target	310
esModuleInterop	310
isolatedModules	311
Strictness Options.	312
noUncheckedIndexedAccess.	312
Other Strictness Options.	314
The Two Choices for module	314
NodeNext	314
Preserve.	315
noEmit	316
Source Maps	316
Transpiling Code for Library Use	317
outDir	317
Creating Declaration Files	317
Declaration Maps	318
jsx	319
A Note on Node's TypeScript Support	320
Managing Multiple TypeScript Configurations	321
How TypeScript Finds tsconfig.json	322
Extending Configurations	323

-project	325
Project References	325
Summary	326

PART VI: ADVANCED APPLICATION DEVELOPMENT 329

15 DESIGNING YOUR TYPES 331

Generic Types	332
Multiple Type Parameters	334
All Type Arguments Must Be Provided	336
Default Type Parameters	336
Type Parameter Constraints	337
Template Literal Types in TypeScript	338
Combining Template Literal Types with Union Types	339
Transforming String Types	339
Conditional Types	340
Mapped Types	341
Key Remapping with as	343
Using Mapped Types with Union Types	344
Exercise 15-1: Creating a DataShape Type Helper	344
Exercise 15-2: Typing PromiseFunc	346
Exercise 15-3: Working with the Result Type	346
Exercise 15-4: Constraining the Result Type	348
Exercise 15-5: A Stricter Omit Type	349
Exercise 15-6: Route Matching	351
Exercise 15-7: Sandwich Permutations	352
Exercise 15-8: Attribute Getters	354
Exercise 15-9: Renaming Keys in a Mapped Type	355
Summary	357

16 BUILDING POWERFUL SHARED UTILITIES 359

Generic Functions	360
Generic Function Type Alias vs. Generic Type	361
Missing or Conflicting Type Arguments	361
There Is No Such Thing as a "Generic"	362
The Problems Generic Functions Solve	363
Debugging the Inferred Type of Generic Functions	364
Type Parameter Defaults	364
Constraining Type Parameters	365
Type Predicates	366
Assertion Functions	368
Function Overloads	369
The Implementation Signature	370
Function Overloads vs. Unions	372

Exercise 16-1: Making a Function Generic	372
Exercise 16-2: Default Type Arguments	374
Exercise 16-3: Inference in Generic Functions.	375
Exercise 16-4: Type Parameter Constraints.	377
Exercise 16-5: Combining Generic Types and Functions	379
Exercise 16-6: Multiple Type Arguments in a Generic Function	382
Exercise 16-7: Assertion Functions.	384
Summary	386
INDEX	387