

Table of Contents

Preface	7
Chapter 1: A Refresher of Objects	1
Object literals	1
Properties	2
Methods	4
Object constructors	6
The Object() constructor	8
Object prototypes	10
Using classes	11
Summary	14
Chapter 2: Diving into OOP Principles	15
OOP principles	16
Is JavaScript Object Oriented?	17
Abstraction and modeling support	17
Association	18
Aggregation	19
Composition	20
OOP principles support	20
Encapsulation	20
Inheritance	23
Polymorphism	25
JavaScript OOP versus classical OOP	29
Summary	30
Chapter 3: Working with Encapsulation and Information Hiding	31
Encapsulation and information hiding	31
Convention-based approach	32
Privacy levels using closure	33
Scope and closure	34
Privacy levels	37
Benefits and drawbacks	38
A meta-closure approach	39
Immediately invoked function expressions	39
Creating a meta-closure with an IIFE	40

Managing isolated private members	42
A definitive solution with WeakMaps	43
Property descriptors	45
Controlling access to public properties	46
Using getters and setters	47
Describing properties	48
Properties with internal state	51
Information hiding in ES6 classes	53
Summary	56
Chapter 4: Inheriting and Creating Mixins	57
<hr/>	
Why inheritance?	57
Objects and prototypes	58
What is a prototype?	59
Creating objects	60
Prototype chaining	62
Inheritance and constructors	64
ES6 inheritance	65
Controlling inheritance	67
Overriding methods	67
Overriding properties	69
Protected members	71
Preventing extensions	72
Implementing multiple inheritance	75
Creating and using mixins	77
Mixing prototypes	77
Mixing classes	79
Summary	81
Chapter 5: Defining Contracts with Duck Typing	83
<hr/>	
Managing dynamic typing	83
Dynamic data types	83
Data typing and objects	85
From data type to instance type	86
Beyond the instance type	87
Contracts and interfaces	89
Duck typing	91
A basic approach	92
A general solution	93
Emulating Interfaces with duck typing	95

Multiple interface implementation	99
Duck typing and polymorphism	100
Summary	102
Chapter 6: Advanced Object Creation	103
<hr/>	
Creating objects	103
Design patterns and object creation	104
Creating a singleton	105
The mysterious behavior of constructors	106
Singletons	107
When to use singletons?	109
An object factory	109
Understanding factories	109
Factory with constructor registration	113
The abstract factory	114
The builder pattern	117
When to use the builder pattern?	119
Comparing factory and builder patterns	120
Recycling objects with an object pool	121
Summary	125
Chapter 7: Presenting Data to the User	127
<hr/>	
Managing user interfaces	127
The user interface problems	128
User interfaces and JavaScript	128
Presentation patterns	131
Model, View, and Controller	131
The Model-View-Controller pattern	132
The Model-View-Presenter pattern	137
The Model-View-ViewModel pattern	143
A MV* pattern comparison	148
Summary	149
Chapter 8: Data Binding	151
<hr/>	
What is data binding?	151
Data binding elements	152
Data binding directions	153
Implementing data binding	154
Manual data binding	154
Monitoring changes	155
Hacking properties	157

Defining a binder	158
The publish/subscribe pattern	160
The observer pattern	160
The publisher/subscriber pattern	161
Implementing observables	162
Using proxies	164
The proxy class	164
Data binding with proxies	166
Summary	168
Chapter 9: Asynchronous Programming and Promises	169
<hr/>	
Is JavaScript asynchronous?	169
Event loop and asynchronous code	170
Events, Ajax, and other asynchronous stuff	171
Writing asynchronous code	171
Using events properties	171
Using callbacks	172
Callbacks and this	173
The callback hell	175
Organizing callbacks	177
The issues of asynchronous code	178
Introducing Promises	179
What are Promises?	179
The Promise terminology	180
Creating Promises	181
Consuming Promises	182
Catching failures	184
Composing Promises	186
Using Generators	190
Introducing Generators	190
Using Generators for asynchronous tasks	191
ES7 async/await	192
Summary	193
Chapter 10: Organizing Code	195
<hr/>	
The global scope	195
Global definitions	196
Creating namespaces	197
Namespaces as object literals	198
Defining namespaces with IIFE	200

The module pattern	201
Modules versus namespaces	201
Using anonymous closures	202
Importing modules	203
Augmenting modules	204
Loose augmentation	205
Overriding a module's methods	205
Tight augmentation	206
Composing modules	207
Submodules	207
Module loading	208
Modules, scripts, and files	208
A simple module loader	210
CommonJS modules	212
Asynchronous Module Definition	213
Merging the module pattern with AMD	215
Universal Module Definition	217
UMD	217
Dependency management	219
ECMAScript 6 modules	219
ES6 module loading	222
Summary	223
Chapter 11: SOLID Principles	225
<hr/>	
Principle of OOP design	225
The Single Responsibility Principle	226
The Open/Closed Principle	230
The Liskov Substitution Principle	235
The Interface Segregation Principle	237
The Dependency Inversion Principle	240
Dependency inversion, inversion of control, and dependency injection	244
Dependency injection approaches	245
Summary	246
Chapter 12: Modern Application Architectures	247
<hr/>	
From scripts to applications	247
What is a large-scale application?	248
What is an application architecture?	249
Goals of an architecture design	250
From old-style to Single Page Applications	250
Old-style web applications	250

Single Page Applications	251
The server role	252
View composition	253
Navigation and routing	254
The remote data	254
The Zakas/Osmani architecture	255
The overall architecture	256
The modules	256
The Sandbox	257
The facade pattern	258
The application core	260
The mediator pattern	261
The base library	263
Cross-cutting features and AOP	263
The log management example	264
Using inheritance	264
The Aspect-Oriented Programming approach	265
Isomorphic applications	266
Summary	267
Index	269
