

Оглавление

Предисловие	14
Вступительное слово	16
Благодарности.....	17
О книге.....	18
Для кого эта книга	18
Структура книги	18
О коде в книге	20
Форум liveBook	20
Об авторе.....	21
Иллюстрации на обложке	22
От издательства	23
Глава 1. Программирование объектов: введение.....	24
1.1. Классы и объекты.....	25
1.2. Состояние	27
1.3. Поведение	30
1.4. Зависимости.....	33
1.5. Наследование.....	34
1.6. Полиморфизм.....	37
1.7. Композиция.....	38
1.8. Организация классов	39

1.9. Оператор возврата и исключения	39
1.10. Модульное тестирование	42
1.11. Динамические массивы	47
Заключение	49
Глава 2. Создание сервисов	50
2.1. Два типа объектов	50
2.2. Внедрение зависимостей и значений конфигурации в качестве аргументов конструктора.....	52
2.2.1. Хранение связанных файлов конфигурации	54
2.3. Внедряйте необходимые сущности, а не место их расположения	56
2.4. Все аргументы конструктора должны быть обязательными	60
2.5. Внедряйте зависимости только в конструкторе.....	62
2.6. Не существует необязательных зависимостей	63
2.7. Делайте все зависимости явными	65
2.7.1. Преобразуйте статические зависимости в зависимости объектов.....	65
2.7.2. Преобразуйте сложные функции в зависимости объектов.....	66
2.7.3. Делайте вызовы системных функций явными	68
2.8. Передавайте данные для выполнения задач как аргументы метода, а не аргументы конструктора.....	72
2.9. После инстанцирования сервиса его поведение должно оставаться неизменным	75
2.10. Не делайте ничего внутри конструктора, кроме инициализации свойств.....	78
2.11. Выдавайте исключение при недопустимом аргументе.....	83
2.12. Объявляйте сервисы как неизменяемые графы объектов с ограниченным рядом точек входа	87
Заклучение	90
Ответы к упражнениям.....	90
Глава 3. Создание других объектов	94
3.1. Запрашивайте минимально необходимый для согласованного поведения объекта объем данных	95

3.2. Запрашивайте только данные, которые имеют смысл.....	97
3.3. Не используйте собственные классы исключений при проверке недопустимых аргументов.....	103
3.4. Проверяйте специфические исключения для недопустимых аргументов, анализируя сообщения исключений	104
3.5. Создавайте новые объекты, чтобы избежать многократной проверки инвариантов предметной области.....	105
3.6. Создавайте новые объекты для представления составных значений.....	108
3.7. Используйте проверки утверждений для аргументов конструктора	110
3.8. Не внедряйте зависимости, а передавайте их в качестве аргументов методов.....	113
3.9. Используйте именованные конструкторы	117
3.9.1. Создавайте объекты из значений примитивного типа.....	117
3.9.2. Не добавляйте без необходимости такие методы, как toString() или toInt()	119
3.9.3. Продумывайте и внедряйте понятия, специфичные для предметной области	119
3.9.4. Для введения ограничения можно использовать приватный конструктор.....	119
3.10. Не используйте заполнители свойств	122
3.11. Добавляйте в объект только то, что нужно.....	122
3.12. Не тестируйте конструкторы.....	123
3.13. Исключение из правила: объекты для передачи данных	127
3.13.1. Используйте публичный модификатор для свойств	127
3.13.2. Не выдавайте исключения, а собирайте ошибки при проверках.....	128
3.13.3. Если нужно, используйте заполнение свойств	129
Заключение	130
Ответы к упражнениям.....	131
Глава 4. Изменение объектов	134
4.1. Сущности: идентифицируемые объекты, которые отслеживают изменения и фиксируют события	135

4.2. Объекты-значения: заменяемые, анонимные и неизменяемые значения.....	138
4.3. Объекты для передачи данных: простые объекты с минимальным набором правил проектирования.....	140
4.4. Отдавайте предпочтение неизменяемым объектам.....	143
4.4.1. Заменяйте значения новыми, а не изменяйте их.....	144
4.5. Модификатор неизменяемого объекта должен возвращать модифицированную копию.....	147
4.6. В изменяемых объектах методы-модификаторы должны быть командными.....	150
4.7. В неизменяемых объектах методы-модификаторы должны иметь декларативные имена.....	151
4.8. Сравните объекты целиком.....	153
4.9. При сравнении неизменяемых объектов проверяйте, что объекты равны, а не одинаковы.....	154
4.10. Вызов метода-модификатора должен всегда оставлять действительный объект.....	156
4.11. Метод-модификатор должен проверять, что запрашиваемое изменение состояния допустимо.....	159
4.12. Используйте запись внутренних событий для проверки изменяемых объектов.....	160
4.13. Не реализуйте текущие интерфейсы в изменяемых объектах.....	166
Заключение.....	170
Ответы к упражнениям.....	170
Глава 5. Использование объектов.....	172
5.1. Шаблон реализации методов.....	172
5.1.1. Проверка предусловий.....	173
5.1.2. Сценарии появления ошибок.....	174
5.1.3. Счастливый путь.....	175
5.1.4. Проверки постусловий.....	176
5.1.5. Возвращаемое значение.....	176
5.2. Некоторые правила для исключений.....	177
5.2.1. Используйте собственные классы исключений только при необходимости.....	177

5.2.2. Именованние недопустимых аргументов или классов логических исключений	178
5.2.3. Именованние классов исключений времени исполнения.....	179
5.2.4. Используйте именованные конструкторы для указания причин ошибки	179
5.2.5. Сопровождайте ошибки подробным описанием.....	179
Заключение	181
Ответы к упражнениям.....	181
Глава 6. Извлечение информации	182
6.1. Используйте методы-запросы для извлечения информации.....	182
6.2. Методы-запросы должны иметь возвращаемые значения единого типа	185
6.3. Избегайте использования методов-запросов, раскрывающих внутренние данные объектов.....	188
6.4. Задавайте специфичные методы и возвращаемые типы для необходимых запросов.....	194
6.5. Задавайте абстракцию для запросов, которые выходят за границы системы.....	196
6.6. Используйте заглушки в тестовых дублерах для методов-запросов	200
6.7. Методы-запросы должны использовать другие методы-запросы, а не командные методы	204
Заключение	207
Ответы к упражнениям.....	207
Глава 7. Выполнение задач.....	209
7.1. Используйте командные методы с именем в императивной форме.....	210
7.2. Ограничивайте область воздействия командного метода и используйте события для выполнения второстепенных задач.....	210
7.3. Создавайте сервис неизменяемым изнутри и снаружи	214
7.4. Когда что-то идет не так, выдавайте исключение.....	218
7.5. Используйте запросы для сбора информации, а командные методы — для последующих действий	219

7.6. Задавайте абстракции для команд, которые выходят за границы системы.....	221
7.7. Проверяйте имитацией (mock) только командные методы	223
Заключение	226
Ответы к упражнениям.....	227
Глава 8. Разделение функций.....	228
8.1. Отделяйте модели записи от моделей чтения.....	229
8.2. Создавайте модели чтения с учетом сценариев их использования.....	236
8.3. Создавайте модели чтения непосредственно из их источника данных.....	238
8.4. Построение моделей чтения из событий предметной области	239
Заключение	244
Ответы к упражнениям.....	245
Глава 9. Изменение поведения сервисов.....	246
9.1. Введите аргументы конструктора, чтобы сделать поведение настраиваемым	247
9.2. Введите аргументы конструктора, чтобы сделать поведение заменяемым.....	248
9.3. Создавайте абстракции, чтобы добиться более сложного поведения	251
9.4. Декорируйте существующее поведение	253
9.5. Используйте объекты уведомлений или прослушватели событий для добавления поведения	256
9.6. Не используйте наследование для изменения поведения объекта.....	261
9.6.1. Когда можно использовать наследование?	264
9.7. Помечайте классы как final по умолчанию	266
9.8. Помечайте методы и свойства как private по умолчанию.....	266
Заключение	268
Ответы к упражнениям.....	269
Глава 10. Справочник объектов.....	273
10.1. Контроллеры	274
10.2. Службы приложений.....	278

12 Оглавление

10.3. Репозитории моделей записи	280
10.4. Сущности	282
10.5. Объекты-значения	283
10.6. Прослушиватели событий	286
10.7. Модели чтения и репозитории моделей чтения	288
10.8. Абстракции, конкретика, слои и зависимости	291
Заключение	293
Глава 11. Эпилог	294
11.1. Архитектурные шаблоны	295
11.2. Тестирование	295
11.2.1. Тестирование класса в сравнении с тестированием объекта	296
11.2.2. Разработка функций сверху вниз	296
11.3. Предметно-ориентированное проектирование	298
11.4. Заключение	298
Приложение. Стандарт кодирования для примеров кода	299