

# ЗМІСТ

ПЕРЕДМОВА.....	13
ВСТУП .....	18
ПОДЯКИ.....	20
РОЗДІЛ 1. ЧИСТИЙ КОД.....	21
Хай живе код.....	22
Поганий код.....	22
Розплата за хаос .....	23
<i>Грандіозне перероблення</i> .....	24
<i>Відомини</i> .....	25
<i>Гігантський парадокс</i> .....	26
<i>Мистецтво чистого коду?</i> .....	26
<i>Що таке «чистий код»?</i> .....	26
Школи думки.....	32
Ми — автори.....	33
Правило бойскаута.....	34
Передісторія й принципи.....	34
Висновки.....	35
Література.....	35
РОЗДІЛ 2. ЗМІСТОВНІ ІМЕНА .....	36
Імена мають передавати наміри програміста.....	37
Уникайте дезінформації .....	38
Використовуйте осмислені відмінності.....	39
Використовуйте імена, що зручно вимовляти.....	41
Вибирайте імена, зручні для пошуку .....	42
Уникайте схем кодування імен.....	42
<i>Угорський запис</i> .....	43
<i>Префікси членів класів</i> .....	43
<i>Інтерфейси та реалізації</i> .....	44
Уникайте ментальних перетворень.....	44
Імена класів .....	45

Купити книгу на сайті [kniga.biz.ua](http://kniga.biz.ua) >>>

Імена методів .....	45
Уникайте дотепів .....	45
Виберіть одне слово для кожної концепції .....	46
Утримуйтеся від каламбурів .....	46
Надавайте імена в термінах рішення .....	47
Надавайте імена в термінах задачі .....	47
Додайте змістовний контекст .....	47
Не додавайте надлишкового контексту .....	49
Кілька слів наостанок .....	50
<b>РОЗДІЛ 3. ФУНКЦІЇ</b> .....	<b>51</b>
Компактність! .....	54
<i>Блоки й відступи</i> .....	55
Правило однієї операції .....	55
<i>Секції у функціях</i> .....	56
Один рівень абстракції на функцію .....	56
<i>Читання коду зорити вище: правило зняження</i> .....	57
Команди <code>switch</code> .....	57
Використовуйте змістовні імена .....	59
Аргументи функцій .....	60
<i>Стандартні унарні форми</i> .....	61
<i>Аргументи-прапори</i> .....	62
<i>Бінарні функції</i> .....	62
<i>Тернарні функції</i> .....	63
<i>Об'єкти як аргументи</i> .....	63
<i>Списки аргументів</i> .....	63
<i>Дієслова й ключові слова</i> .....	64
Позабантеся побічних ефектів .....	64
<i>Вихідні аргументи</i> .....	65
Поділ команд і запитів .....	66
Використовуйте вищатики замість повернення кодів помилок .....	67
<i>Виділіть блоки <code>try/catch</code></i> .....	67
<i>Оброблення помилок як одна операція</i> .....	68
<i>Маліміт залежностей <code>Error.java</code></i> .....	68
Не повторюйтеся .....	69
Структурне програмування .....	69
Як навчитися писати такі функції? .....	70
Висновок .....	70
Література .....	73

<b>РОЗДІЛ 4. КОМЕНТАРІ</b> .....	74
Коментарі не компенсують поганого коду .....	76
Поясніть свої наміри в коді .....	76
Гарні коментарі .....	76
Юридичні коментарі .....	77
Інформативні коментарі .....	77
Презентування намірів .....	77
Проксметки .....	78
Попередження про наслідки .....	79
Коментарі TODO .....	80
Посилення .....	80
Коментарі Javadoc у загальнодоступних API .....	81
Погані коментарі .....	81
Бурмотиння .....	81
Надлишкові коментарі .....	82
Недостовірні коментарі .....	84
Обов'язкові коментарі .....	85
Журнальні коментарі .....	85
Шум .....	86
Небезпечний шум .....	88
Не використовуйте коментарі там, де можна використати функцію або зміну .....	89
Позиційні маркери .....	89
Коментарі за закритою фігурною дужкою .....	89
Посилання на авторів .....	90
Закоментований код .....	90
HTML коментарі .....	91
Неповнальна інформація .....	92
Занадто багато інформації .....	92
Неочевидні коментарі .....	93
Заголовки функцій .....	93
Заголовки Javadoc у внутрішньому коді .....	93
Приклад .....	93
Література .....	97
<b>РОЗДІЛ 5. ФОРМАТУВАННЯ</b> .....	98
Мета форматування .....	99
Вертикальне форматування .....	99
Газетна метафора .....	100
Вертикальний розподіл коментарів .....	101
Вертикальне стиснення .....	102
Вертикальні відстані .....	103
Вертикальне впорядкування .....	108

Горизонтальне форматування .....	108
<i>Горизонтальний розподіл і стиснення</i> .....	109
<i>Горизонтальне вирівнювання</i> .....	110
<i>Відступи</i> .....	111
<i>Вироджені ділянки видимості</i> .....	113
Правила форматування в командах .....	113
Правила форматування від дядечка Боба .....	114
<b>РОЗДІЛ 6. ОБ'ЄКТИ Й СТРУКТУРИ ДАНИХ</b> .....	<b>117</b>
Абстракція даних .....	118
Антисиметрія даних/об'єктів .....	119
Закон Деметри .....	122
<i>«Аварія потяга»</i> .....	122
<i>Гібриди</i> .....	123
<i>Приховування структури</i> .....	123
Об'єкти передавання даних .....	124
<i>Активні записи</i> .....	125
Висновки .....	126
Література .....	126
<b>РОЗДІЛ 7. ОБРОБЛЕННЯ ПОМИЛОК (Майк Фізерс)</b> .....	<b>127</b>
Використовуйте винятки замість кодів помилок .....	128
Почніть із написання команди <code>try-catch-finally</code> .....	129
Використовуйте неперевірні винятки .....	131
Передавайте контекст із винятками .....	132
Визначайте класи винятків у контексті потреб виключної сторони .....	132
Визначте нормальний шлях виконання .....	134
Не повертайте <code>null</code> .....	135
Не передавайте <code>null</code> .....	136
Висновки .....	138
Література .....	138
<b>РОЗДІЛ 8. МЕЖІ</b> .....	<b>139</b>
Використання стороннього коду .....	140
Дослідження та аналіз меж .....	142
Вивчення <code>log4j</code> .....	142
Навчальні тести: вигідніше, ніж безкоштовно .....	144
Використання неіснуючого коду .....	145

Чисті межі.....	146
Література.....	146
<b>РОЗДІЛ 9. МОДУЛЬНІ ТЕСТИ.....</b>	<b>147</b>
Три закони TDD.....	148
Про чистоту тестів.....	149
<i>Тести як засіб забезпечення змін</i> .....	150
Чисті тести.....	151
<i>Предметно-орієнтована мова тестування</i> .....	153
<i>Подвійний стандарт</i> .....	154
Одна перевірка на тест.....	156
<i>Одна концепція на тест</i> .....	157
FIRST.....	159
Висновки.....	160
Література.....	160
<b>РОЗДІЛ 10. КЛАСИ (спільно з Джеффри Лангром).....</b>	<b>161</b>
Будова класу.....	161
<i>Інкапсуляція</i> .....	162
Класи мають бути компактними!.....	162
<i>Принцип єдиної відповідальності (SRP)</i> .....	164
<i>Зв'язність</i> .....	166
<i>Підтримання зв'язності призводить до зменшення класів</i> .....	167
Структурування з урахуванням змін.....	173
<i>Ізоляція змін</i> .....	176
Література.....	178
<b>РОЗДІЛ 11. СИСТЕМИ (Кевін Дін Вомплер).....</b>	<b>179</b>
Як би ви будували місто?.....	179
Відокремлення конструювання системи від її використання.....	180
<i>Відокремлення main</i> .....	181
<i>Фабрики</i> .....	182
<i>Впровадження залежностей</i> .....	182
Масштабування.....	183
<i>Перехресні ділянки відповідальності</i> .....	186
Посередники.....	187
АОП-інфраструктури «чистою» Java.....	189
Аспекти <i>Azres!</i> .....	192
Випробування системної архітектури.....	193
Оптимізація прийняття рішень.....	194

Застосуйте стандарти розумно, коли вони приносять очевидну користь.....	194
Системам необхідні предметно-орієнтовані мови.....	195
Висновки.....	195
Література.....	196
<b>РОЗДІЛ 12. ФОРМУВАННЯ АРХІТЕКТУРИ (Джефф Лангр).....</b>	<b>197</b>
Чотири правила.....	197
Правило № 1: виконання всіх тестів.....	198
Правила № 2–4: перероблення коду.....	198
Відсутність дублювання.....	199
Виразність.....	201
Мінімум класів і методів.....	202
Висновки.....	203
Література.....	203
<b>РОЗДІЛ 13. БАГАТОПОТОКОВІСТЬ (Бретт Л. Шухерт).....</b>	<b>204</b>
Навіщо потрібна багатопотоковість?.....	205
Міфи й неправильні уявлення.....	206
Труднощі.....	207
Захист від помилок багатопотоковості.....	207
Принцип єдиної відповідальності.....	208
Наслідки: обмежуйте ділянку видимості даних.....	208
Наслідки: використовуйте копії даних.....	208
Наслідки: потоки мають бути якомога більш незалежними.....	209
Знайте свою бібліотеку.....	209
Потоково-безпечні колекції.....	209
Знайте моделі виконання.....	210
Модель «виробник / споживач».....	211
Модель «читач / письменник».....	211
Модель «філософи за обідом».....	211
Остерігайтеся залежностей між синхронізованими методами.....	212
Синхронізовані секції повинні мати мінімальний розмір.....	213
Про труднощі коректного завершення.....	213
Тестування багатопотокового коду.....	214
Розглядайте неперіодичні збої як ознаки можливих проблем багатопотоковості.....	214
Почніть із налагодження основного коду, не пов'язаного з багатопотоковістю.....	214
Реалізуйте перемикання конфігурацій багатопотокового коду.....	215

<i>Забезпечте логічну ізоляцію конфігурацій багатопотокового коду</i> .....	215
<i>Протестуйте програму з кількістю потоків, що перевищує кількість процесорів</i> .....	215
<i>Протестуйте програму на різних платформах</i> .....	216
<i>Застосовуйте інструментування коду для підвищення ймовірності виявлення збоїв</i> .....	216
<i>Ручне інструментування</i> .....	216
<i>Автоматизоване інструментування</i> .....	217
Висновки.....	218
Література.....	219
<b>РОЗДІЛ 14. ПОСЛІДОВНЕ ОЧИЩЕННЯ</b> (Справа про розбирання аргументів командного рядка) .....	220
<i>Реалізація Args</i> .....	221
<i>Як я це зробив?</i> .....	227
<i>Args: чернетка</i> .....	228
<i>На цьому я зупинився</i> .....	240
<i>Про наступове вдосконалення</i> .....	240
<i>Аргументи String</i> .....	242
Висновки .....	281
<b>РОЗДІЛ 15. ВНУТРІШНЯ БУДОВА JUNIT</b> .....	282
Фреймворк JUnit .....	282
Висновки.....	297
<b>РОЗДІЛ 16. ПЕРЕРОБЛЕННЯ SERIALDATE</b> .....	298
Перш за все — змусити працювати .....	299
...Потім очистити код.....	301
Висновки .....	315
Література.....	315
<b>РОЗДІЛ 17. «ЗАПАХИ» ТА ЕВРИСТИЧНІ ПРАВИЛА</b> .....	316
Коментарі .....	317
<i>C1: Недоречна інформація</i> .....	317
<i>C2: Застарілий коментар</i> .....	317
<i>C3: Надмірний коментар</i> .....	317
<i>C4: Погано написаний коментар</i> .....	318
<i>C5: Закоментований код</i> .....	318
Робоче середовище.....	318
<i>E1: Збирання складається з декількох етапів</i> .....	318
<i>E2: Тестування складається з декількох етапів</i> .....	318

Функції	319
F1: Забагато аргументів	319
F2: Вихідні аргументи	319
F3: Прапори в аргументах	319
F4: Мертві функції	319
Різне	319
G1: Кілька мов в одному вихідному файлі	319
G2: Очевидна поведінка не реалізована	319
G3: Некоректна межова поведінка	320
G4: Відмікнені засоби безпеки	320
G5: Дублювання	320
G6: Код на неправильному рівні абстракції	321
G7: Базові класи, залежні від похідних	322
G8: Забагато інформації	322
G9: Мертвий код	323
G10: Вертикальний розподіл	323
G11: Непослідовність	323
G12: Баласт	324
G13: Штучні прив'язки	324
G14: Функціональна заздрість	324
G15: Аргументи-селектори	325
G16: Незрозумілі наміри	326
G17: Неправильне розміщення	326
G18: Недоречні статичні методи	327
G19: Використовуйте повсюдовальні зміни	327
G20: Імена функцій повинні описувати виконувану операцію	328
G21: Розуважити алгоритму	328
G22: Перетворення логічних залежностей на фізичні	329
G23: Використовуйте поліморфізм замість if/else або switch/case	330
G24: Дотримуйте стандартних конвенцій	331
G25: Заміняйте «чарівні числа» іменованими константами	331
G26: Будьте точні	332
G27: Структура важливіша за конвенції	333
G28: Інкапсулюйте умовні конструкції	333
G29: Уникайте негативних умов	333
G30: Функції мають виконувати одну операцію	333
G31: Приховані темпоральні прив'язки	334
G32: Структура коду має бути об'єднана	335
G33: Інкапсулюйте межові умови	336
G34: Функції мають бути матисані на одному рівні абстракції	336
G35: Звертайте конфігураційні дані на високих рівнях	338
G36: Уникайте тривалих звернень	338



Замі	339
J1: Використовуйте узагальнені директиви імпорту	339
J2: Не успадковуйте від констант	340
J3: Константи проти перерахувань	341
Імена	342
N1: Використовуйте іменовані імена	342
N2: Вибирайте імена на відповідному рівні абстракції	343
N3: За можливості використовуйте стандартну номенклатуру	344
N4: Недвозначні імена	344
N5: Користуйтеся довгими іменами для довгих зон видимості	345
N6: Уникайте кодування	345
N7: Імена повинні описувати побічні ефекти	345
Тести	346
T1: Брик тестів	346
T2: Використовуйте засоби аналізу покриття коду	346
T3: Не пропускайте тривіальні тести	346
T4: Відіокремляйте тести як підання	346
T5: Тестуйте межові умови	346
T6: Ретельно тестуйте код поруч із помилками	346
T7: Закономірності збоїв часто несуть корисну інформацію	347
T8: Закономірності покриття коду часто несуть корисну інформацію	347
T9: Тести повинні працювати швидко	347
Висновки	347
Література	347
<b>ДОДАТОК А. БАГАТОПОТОКОВІСТЬ II (Бретт Л. Шухерт)</b>	<b>348</b>
Приклад застосунок «клієнт/сервер»	348
Сервер	348
Реалізація багатопотоковості	350
Аналіз серверного коду	350
Висновки	352
Можливі шляхи виконання	353
Кількість шляхів	353
Обчислення можливих варіантів утвордження	354
Копіємо глибше	355
Висновки	358
Знайте свої бібліотеки	358
Elasticor Framework	358
Неблокові рішення	359
Потоково-небезпечні класи	360

Залежності між методами можуть порушити роботу багатопотокового коду .....	361
Перенесення збоїв .....	362
Клієнтське блокування .....	362
Серверне блокування .....	364
Підвищення продуктивності .....	365
Обчислення продуктивності в однопотоковій моделі .....	367
Обчислення продуктивності в багатопотоковій моделі .....	367
Взаємне блокування .....	368
Взаємне виключення .....	369
Блокування з очікуванням .....	369
Відсутність виніснення .....	369
Циклічне очікування .....	369
Порушення вземного виключення .....	370
Порушення блокування з очікуванням .....	370
Порушення відсутності виніснення .....	370
Порушення циклічного очікування .....	371
Тестування багатопотокового коду .....	371
Засоби тестування багатопотокового коду .....	375
Висновки .....	375
Повні приклади коду .....	376
Однопотокова реалізація архітектури «клієнт/сервер» .....	376
Архітектура «клієнт/сервер» з використанням потоків .....	379
<b>ДОДАТОК В. <code>ORG.JFREE.DATE.SERIALDATE</code> .....</b>	<b>381</b>
<b>ДОДАТОК С. ПЕРЕХРЕСНІ ПОСИЛАННЯ .....</b>	<b>439</b>
<b>ЕПЛОГ .....</b>	<b>440</b>
<b>ПОКАЖЧИК .....</b>	<b>441</b>