

Предисловие

Предисловие автора ко второй редакции

Предисловие автора ко второй редакции

Введение

Создание и уничтожение объектов

Рассмотрите возможность замены конструкторов статическими методами генерации

Используйте шаблон Builder, когда приходится иметь дело с большим количеством параметров конструктора

Свойство singleton обеспечивайте закрытым конструктором или типом перечислений

Отсутствие экземпляров обеспечивает закрытый конструктор

Избегайте ненужных объектов

Уничтожайте устаревшие ссылки (на объекты)

Остерегайтесь методов finalize

Методы, общие для всех объектов

Переопределяя метод equals, соблюдайте общие соглашения

Переопределяя метод equals, всегда переопределяйте hashCode

Всегда переопределяйте метод toString

Соблюдайте осторожность при переопределении метода clone

Подумайте над реализацией интерфейса Comparable

Классы и интерфейсы

Сводите к минимуму доступность классов и членов

В открытых классах используйте методы доступа, а не открытые поля

Предпочитайте постоянство

Предпочитайте компоновку наследованию

Проектируйте и документируйте наследование либо запрещайте его

Предпочитайте интерфейсы абстрактным классам

Используйте интерфейсы только для определения типов

Объединение заменяйте иерархией классов

Используйте объект функции для выполнения сравнения

Предпочитайте статические классы-члены нестатическим

Средства обобщенного программирования (Generics)

Не используйте необработанные типы в новом коде

Избегайте предупреждений о непроверенном коде

Предпочитайте списки массивам

Поддерживайте обобщенные типы

Поддерживайте обобщенные методы

Используйте ограниченные групповые символы для увеличения гибкости API

Использование неоднородных контейнеров

Перечислимые типы и аннотации

Используйте перечислимые типы вместо констант

Используйте поля экземпляра вместо числовых значений

Используйте EnumSet вместо битовых полей

Используйте EnumMap вместо порядкового индексирования

Имитируйте расширяемые перечислимые типы с помощью интерфейсов

Предпочитайте аннотации шаблонам присвоения имен

Используйте аннотацию Override последовательно

Используйте маркерные интерфейсы для определения типов

Методы

Проверяйте достоверность параметров

При необходимости создавайте резервные копии

Тщательно проектируйте сигнатуру метода

Перезагружая методы, соблюдайте осторожность

Используйте varargs с осторожностью

Возвращайте массив нулевой длины, а не null  
Для всех открытых элементов API пишите doc-комментарии  
Общие вопросы программирования  
Сводите к минимуму область видимости локальных переменных  
Предпочитайте использование цикла for-each  
Изучите библиотеки и пользуйтесь ими  
Если требуются точные ответы, избегайте использования типов float и double  
Отдавайте предпочтение использованию обычных примитивных типов, а не упакованных примитивных типов  
Не используйте строку там, где более уместен иной тип  
При конкатенации строк опасайтесь потери производительности  
Для ссылки на объект используйте его интерфейс  
Предпочитайте интерфейс отражению класса  
Соблюдайте осторожность при использовании машинозависимых методов  
Соблюдайте осторожность при оптимизации  
При выборе имен придерживайтесь общепринятых соглашений  
Исключения  
Используйте исключения лишь в исключительных ситуациях  
Применяйте обрабатываемые исключения для восстановления, для программных ошибок используйте исключения времени выполнения  
Избегайте ненужных обрабатываемых исключений  
Предпочитайте стандартные исключения  
Иницилируйте исключения, соответствующие абстракции  
Для каждого метода документируйте все иницилируемые исключения  
В описании исключения добавляйте информацию о сбое  
Добивайтесь атомарности методов по отношению к сбоям  
Не игнорируйте исключений  
Потоки  
Синхронизируйте доступ потоков к совместно используемым изменяемым данным.  
Избегайте избыточной синхронизации  
Предпочитайте использование экзекуторов и заданий вместо потоков  
Предпочитайте использовать утилиты параллельности, нежели wait и notify  
При работе с потоками документируйте уровень безопасности  
С осторожностью используйте отложенную инициализацию  
Не попадайте в зависимость от планировщика потоков  
Избегайте группировки потоков  
Сериализация  
Соблюдайте осторожность при реализации интерфейса Serializable  
Рассмотрите возможность использования специализированной сериализованной формы  
Метод readObject должен создаваться с защитой  
Для контроля над экземплярами предпочитайте использование перечислимых типов методу readResolve  
Рассмотрите использование агентов сериализации вместо сериализованных экземпляров  
Список литературы