



## Новые инструменты

Два новых инструмента — два полезных способа работы с коллекциями объектов.

### Принципы

Инкапсулируйте то, что изменяется.  
Предпочитайте композицию наследованию.  
Программируйте на уровне интерфейсов.  
Стремитесь к слабой связности взаимодействующих объектов.  
Классы должны быть открыты для расширения, но закрыты для изменения.  
Код должен зависеть от абстракций, а не от конкретных классов.  
Взаимодействуйте только с «друзьями».  
Не вызывайте нас — мы вас сами вызовем.  
Класс должен иметь только одну причину для изменений.

### Концепции

Абстракция  
Инкапсуляция  
Полиморфизм  
Наследование

Еще один важный принцип, относящийся к изменениям в архитектуре.

### Паттерны

Итератор — предоставляет механизм последовательного перебора элементов коллекции без раскрытия ее внутренней структуры.

Компоновщик — объединяет объекты в древовидные структуры для представления иерархий «часть-целое».

Еще одна глава «два в одном».

## КЛЮЧЕВЫЕ МОМЕНТЫ



- Итератор предоставляет доступ к элементам коллекции без раскрытия ее внутренней структуры.
- Итератор обеспечивает перебор элементов коллекции и его инкапсуляцию в отдельном объекте.
- При использовании итераторов коллекция избавляется от одной обязанности (поддержки операций перебора данных).
- Итератор предоставляет общий интерфейс перебора элементов коллекции, что позволяет применять полиморфизм в коде, использующем элементы коллекции.
- Каждому классу следует по возможности назначать только одну обязанность.
- Паттерн Компоновщик предоставляет структуру для хранения как отдельных объектов, так и комбинаций.
- Паттерн Компоновщик позволяет клиенту выполнять однородные операции с комбинациями и отдельными объектами.
- В реализации паттерна Компоновщик приходится искать баланс между прозрачностью, безопасностью и вашими потребностями.