



## Новые инструменты

Подошла к концу глава 2. В ваш ОО-инструментарий добавились новые инструменты...

### Концепции

#### Абстракция

яция  
физм  
ание

### Принципы

Инкапсулируйте то, что изменяется.

Предпочитайте композицию наследованию.

Программируйте на уровне интерфейсов.

Стремитесь к слабой связанности взаимодействующих объектов.

Новый принцип. Помните: слабосвязанные структуры более гибкие и лучше выдерживают изменения.

### Паттерны

Стратегия — определяет

Наблюдатель — определяет отношение «один-ко-многим» между объектами таким образом, что при изменении состояния одного объекта происходит автоматическое оповещение и обновление всех зависящих объектов

Новый паттерн для слабосвязанного оповещения групп объектов об изменении состояния. Мы еще вернемся к паттерну Наблюдатель, когда речь пойдет о MVC!

## КЛЮЧЕВЫЕ МОМЕНТЫ



- Паттерн Наблюдатель определяет отношение «один-ко-многим» между объектами.
- Субъекты обновляют наблюдателей через единый интерфейс.
- Субъект ничего не знает о наблюдателях — кроме того, что они реализуют интерфейс Observer.
- При использовании паттерна возможен как запрос, так и активная доставка данных от субъекта (запрос считается более «правильным»).
- Работа кода не должна зависеть от порядка оповещения наблюдателей.
- Java содержит несколько реализаций паттерна Наблюдатель, включая обобщенную реализацию `java.util.Observable`.
- Помните о недостатках `java.util.Observable`.
- Не бойтесь самостоятельно реализовать Observable при необходимости.
- Swing, как и многие GUI-инфраструктуры, широко применяют паттерн Наблюдатель.
- Паттерн также встречается во многих других местах, включая JavaBeans и RMI.