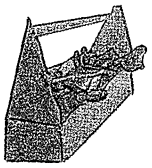
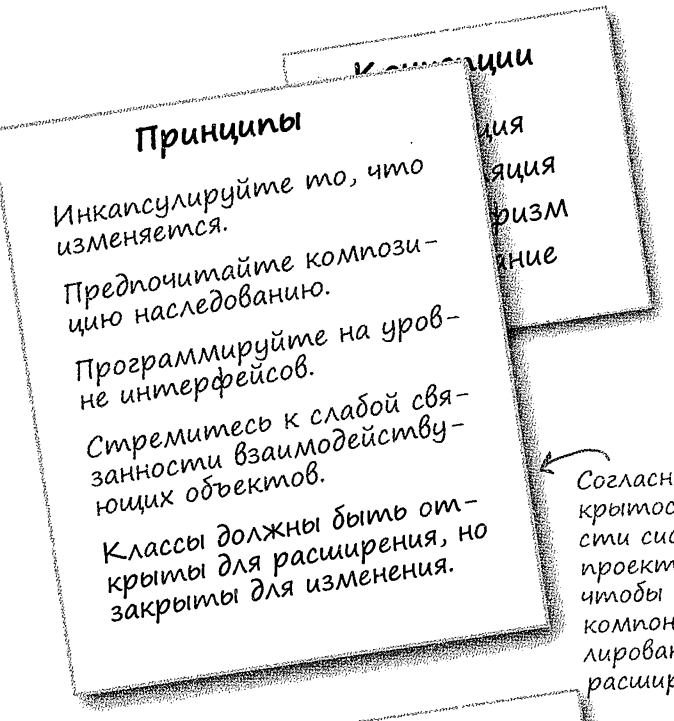
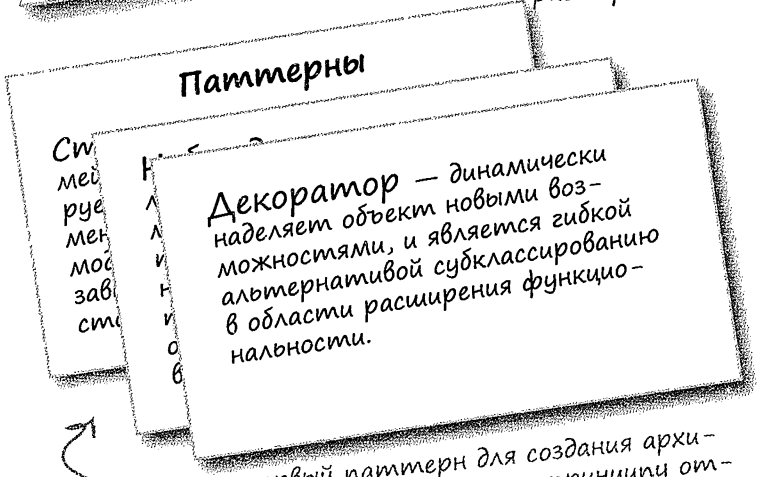


Новые инструменты

Подошла к концу очередная глава, а в вашем ОО-инструментарии появился новый принцип и паттерн.



Согласно принципу открытости/закрытости системы должны проектироваться так, чтобы их закрытые компоненты были изолированы от новых расширений.



Наш первый паттерн для создания архитектур, удовлетворяющих принципу открытости/закрытости. Или не первый? А может, один из описанных ранее паттернов тоже следовал этому принципу?

КЛЮЧЕВЫЕ МОМЕНТЫ

- Наследование — одна из форм расширения, но оно не всегда обеспечивает гибкость архитектуры.
- Следует предусмотреть возможность расширения поведения без изменения существующего кода.
- Композиция и делегирование часто используются для динамического добавления нового поведения.
- Паттерн Декоратор предоставляет альтернативу субклассированию в области расширения поведения.
- Типы декораторов соответствуют типам декорируемых компонентов (соответствие достигается посредством наследования или реализации интерфейса).
- Декораторы изменяют поведение компонентов, добавляя новую функциональность до и (или) после (или даже вместо) вызовов методов компонентов.
- Компонент может декорироваться любым количеством декораторов.
- Декораторы обычно прозрачны для клиентов компонента (если клиентский код не зависит от конкретного типа компонента).