- *посмотрите, что в вашем дизайне должно быть изменяющимся.* Такой подход противоположен исследованию причин, вызвавших необходимость перепроектирования. Вместо этого подумайте, что могло бы *заставить* изменить дизайн, а также о том, что бы вы хотели изменять без перепроектирования. Акцент здесь делается на *инкапсуляции сущностей, подверженных изменениям*, а это предмет многих паттернов. В таблице 1.2 перечислены те аспекты дизайна, которые разные паттерны позволяют варьировать независимо, устраняя тем самым необходимость в перепроектировании.

1.8. Как пользоваться паттерном проектирования

Как пользоваться паттерном проектирования, который вы выбрали для изучения и работы? Вот перечень шагов, которые помогут вам эффективно применить паттерн:

1. *Прочитайте описание паттерна, чтобы получить о нем общее представление.* Особое внимание обратите на разделы «Применимость» и «Результаты» – убедитесь, что выбранный вами паттерн действительно подходит для решения ваших задач.
2. *Вернитесь назад и изучите разделы «Структура», «Участники» и «Отношения».* Убедитесь, что понимаете упоминаемые в паттерне классы и объекты и то, как они взаимодействуют друг с другом.
3. *Посмотрите на раздел «Пример кода», где приведен конкретный пример использования паттерна в программе.* Изучение кода поможет понять, как нужно реализовывать паттерн.
4. *Выберите для участников паттерна подходящие имена.* Имена участников паттерна обычно слишком абстрактны, чтобы употреблять их непосредственно в коде. Тем не менее бывает полезно включить имя участника как имя в программе. Это помогает сделать паттерн более очевидным при реализации. Например, если вы пользуетесь паттерном стратегия в алгоритме размещения текста, то классы могли бы называться `SimpleLayoutStrategy` или `TeXLayoutStrategy`.
5. *Определите классы.* Объявите их интерфейсы, установите отношения наследования и определите переменные экземпляра, которыми будут представлены данные объекты и ссылки на другие объекты. Выявите имеющиеся в вашем приложении классы, на которые паттерн оказывает влияние, и соответствующим образом модифицируйте их.
6. *Определите имена операций, встречающихся в паттерне.* Здесь, как и в предыдущем случае, имена обычно зависят от приложения. Руководствуйтесь теми функциями и взаимодействиями, которые ассоциированы с каждой операцией. Кроме того, будьте последовательны при выборе имен. Например, для обозначения фабричного метода можно было бы всюду использовать префикс `Create-`.
7. *Реализуйте операции, которые выполняют обязанности и отвечают за отношения, определенные в паттерне.* Советы о том, как это лучше сделать, вы найдете в разделе «Реализация». Поможет и «Пример кода».

Все вышесказанное – обычные рекомендации. Со временем вы выработаете собственный подход к работе с паттернами проектирования.

Таблица 1.2. Изменяемые паттернами элементы дизайна

Назначение	Паттерн проектирования	Аспекты, которые можно изменять
Порождающие паттерны	Абстрактная фабрика	Семейства порождаемых объектов
	Одиночка	Единственный экземпляр класса
	Прототип	Класс, из которого инстанцируется объект
	Строитель	Способ создания составного объекта
	Фабричный метод	Инстанцируемый подкласс объекта
Структурные паттерны	Адаптер	Интерфейс к объекту
	Декоратор	Обязанности объекта без порождения подкласса
	Заместитель	Способ доступа к объекту, его местоположение
	Компоновщик	Структура и состав объекта
	Мост	Реализация объекта
	Приспособленец	Накладные расходы на хранение объектов
	Фасад	Интерфейс к подсистеме
Паттерны поведения	Интерпретатор	Грамматика и интерпретация языка
	Итератор	Способ обхода элементов агрегата
	Команда	Время и способ выполнения запроса
	Наблюдатель	Множество объектов, зависящих от другого объекта; способ, которым зависимые объекты поддерживают себя в актуальном состоянии
	Посетитель	Операции, которые можно применить к объекту или объектам, не меняя класса
	Посредник	Объекты, взаимодействующие между собой, и способ их кооперации
	Состояние	Состояние объекта
	Стратегия	Алгоритм
	Хранитель	Закрытая информация, хранящаяся вне объекта, и время ее сохранения
	Цепочка обязанностей	Объект, выполняющий запрос
	Шаблонный метод	Шаги алгоритма

Никакое обсуждение того, как пользоваться паттернами проектирования, нельзя считать полным, если не сказать о том, как не надо их применять. Нередко за гибкость и простоту изменения, которые дают паттерны, приходится платить усложнением дизайна и ухудшением производительности. Паттерн проектирования стоит применять, только когда дополнительная гибкость действительно необходима. Для оценки достоинств и недостатков паттерна большую помощь могут оказать разделы каталога «Результаты».