# Architectural pattern

Паттерны в отличие от архитектур не зависят от моды и других факторов и регулярно появляются при проектировании ИС.

## Big Ball of Mud

## Client/Server

# Structural Design Pattern

## Приспособленец (Flyweight)

### 📌 Назначение (Intent):

Использовать разделение общего состояния (sharing), чтобы эффективно поддерживать большое количество мелких объектов.

### 🎯 Мотивация:

Некоторые приложения выигрывают от использования объектно-ориентированного подхода на всех уровнях, но наивная реализация может оказаться слишком затратной по памяти и производительности.

Пример — текстовый редактор:s

* Объекты могут использоваться для представления встроенных элементов (таблиц, изображений и т.п.).
* Однако обычно не создаётся отдельный объект для каждого символа, так как это слишком тяжело.
* Если бы каждый символ был объектом, это позволило бы:  
  + обрабатывать символы и элементы единообразно (например, при отрисовке и форматировании),
  + гибко расширять систему, добавляя новые языки или шрифты, не затрагивая остальной код,
  + сделать структуру объектов аналогичной структуре документа.

### 📌 Суть подхода:

* Вместо создания миллиона одинаковых объектов (например, каждой буквы "а"), создаём один общий (разделяемый) объект и много "контекстов", которые определяют, где он используется.
* Это особенно полезно в ситуациях с огромным числом однотипных элементов, где можно отделить:  
  + внутреннее (разделяемое) состояние — общее для всех,
  + внешнее (уникальное) состояние — задаётся извне и не хранится внутри объекта.

### 🧠 Пример:

* В документе 100 000 символов, но только 100 уникальных глифов (букв, знаков).
* Вместо 100 000 объектов — храним 100 общих (разделяемых) объектов + координаты каждого символа.

Если хочешь, могу включить также UML-диаграмму и список участников паттерна.