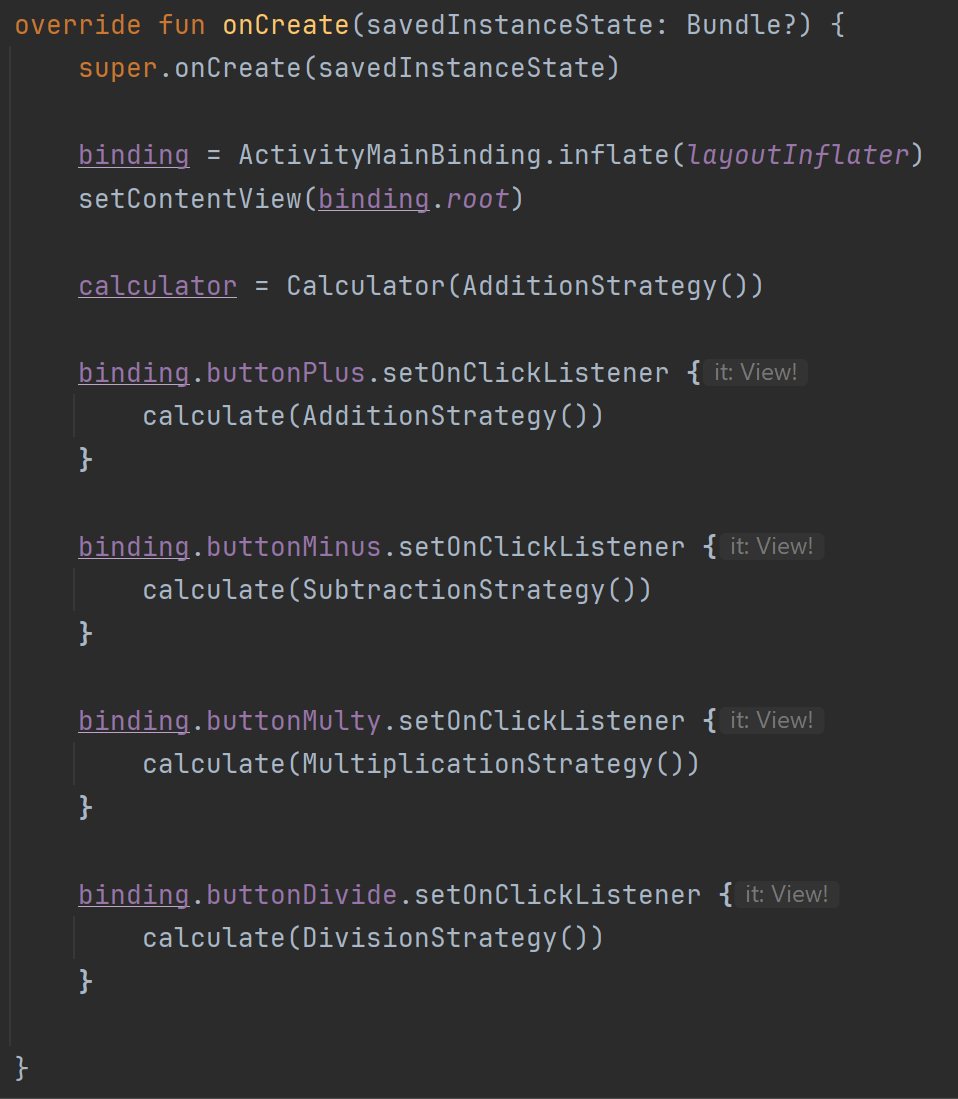
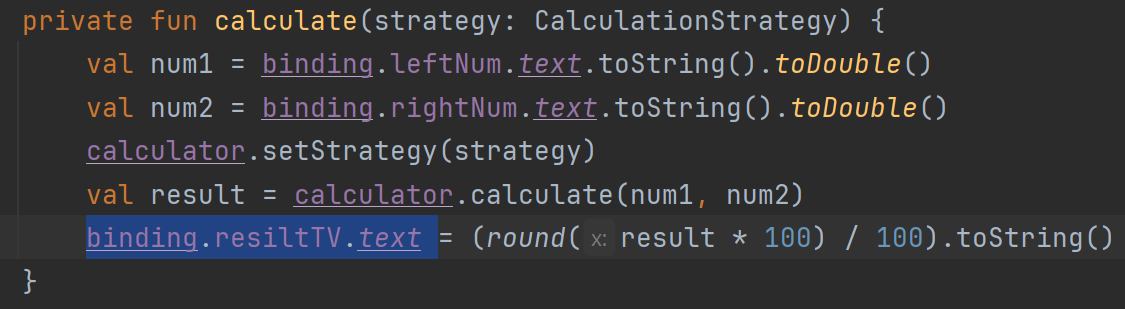
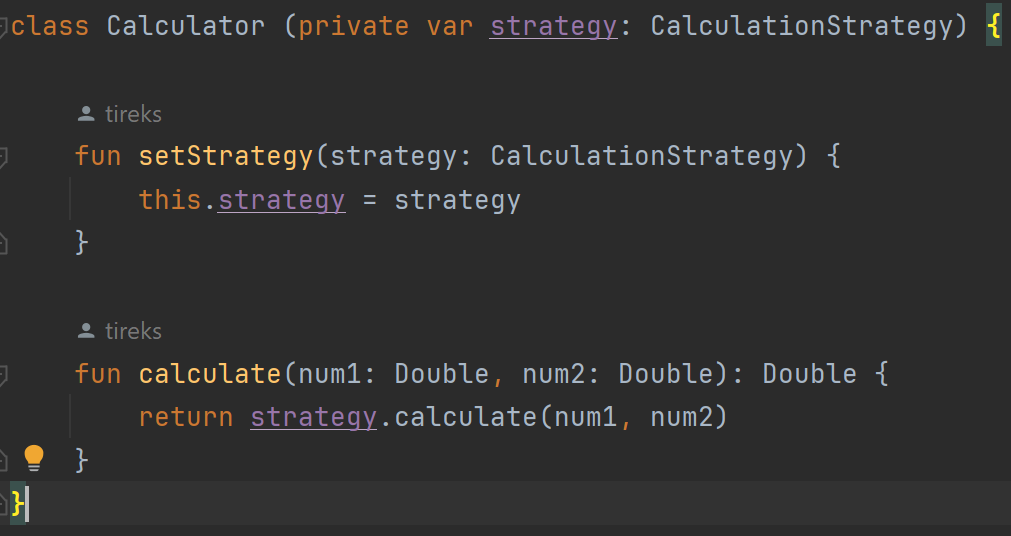
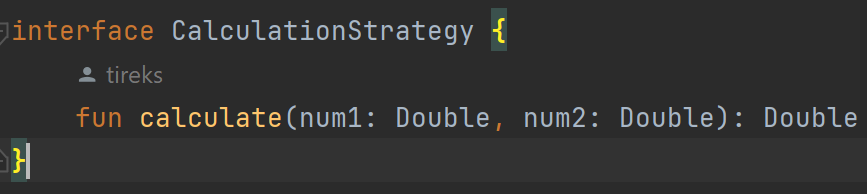
Приложение с применением паттерна Стратегия:

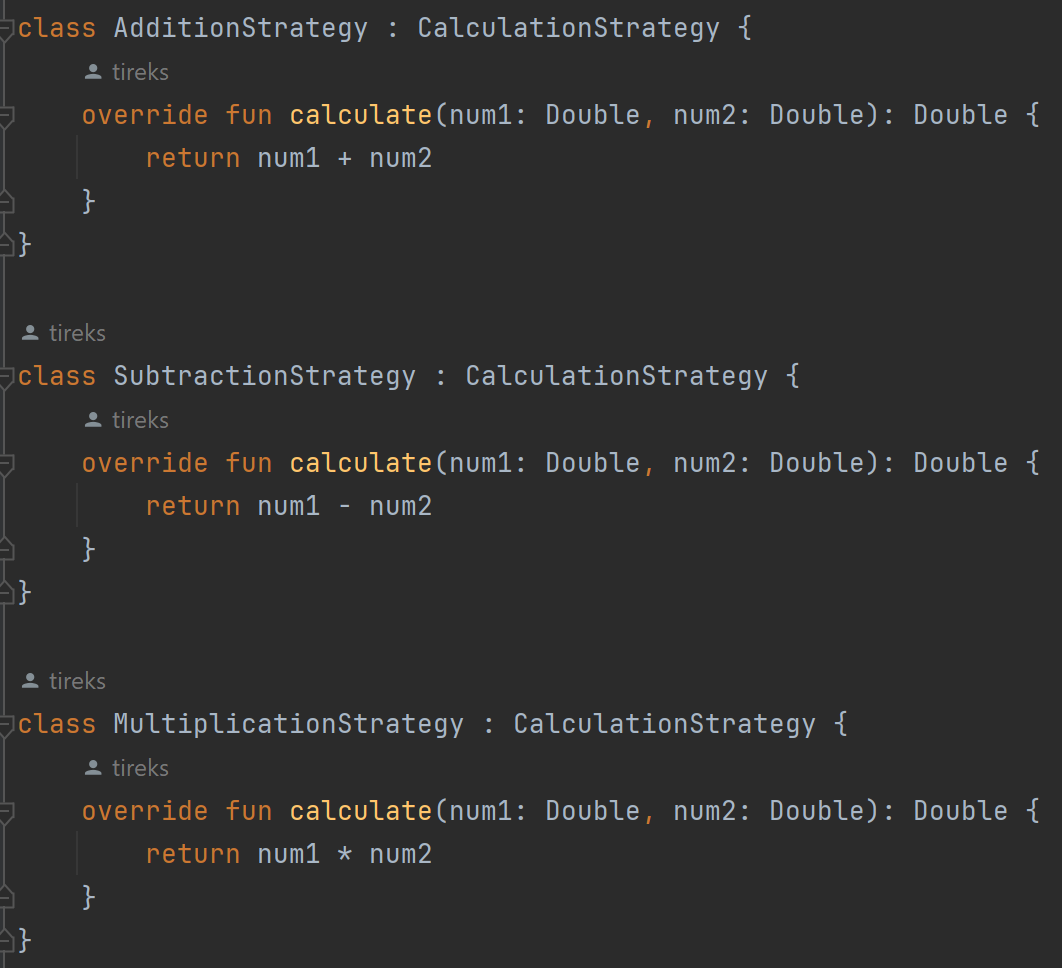
Это приложение представляет собой простой калькулятор, который позволяет пользователю выбирать различные операции (сложение, вычитание, умножение, деление) с помощью паттерна Стратегия. Основная идея заключается в том, что каждая операция представлена отдельной стратегией, что позволяет легко добавлять новые операции без изменения основного кода.





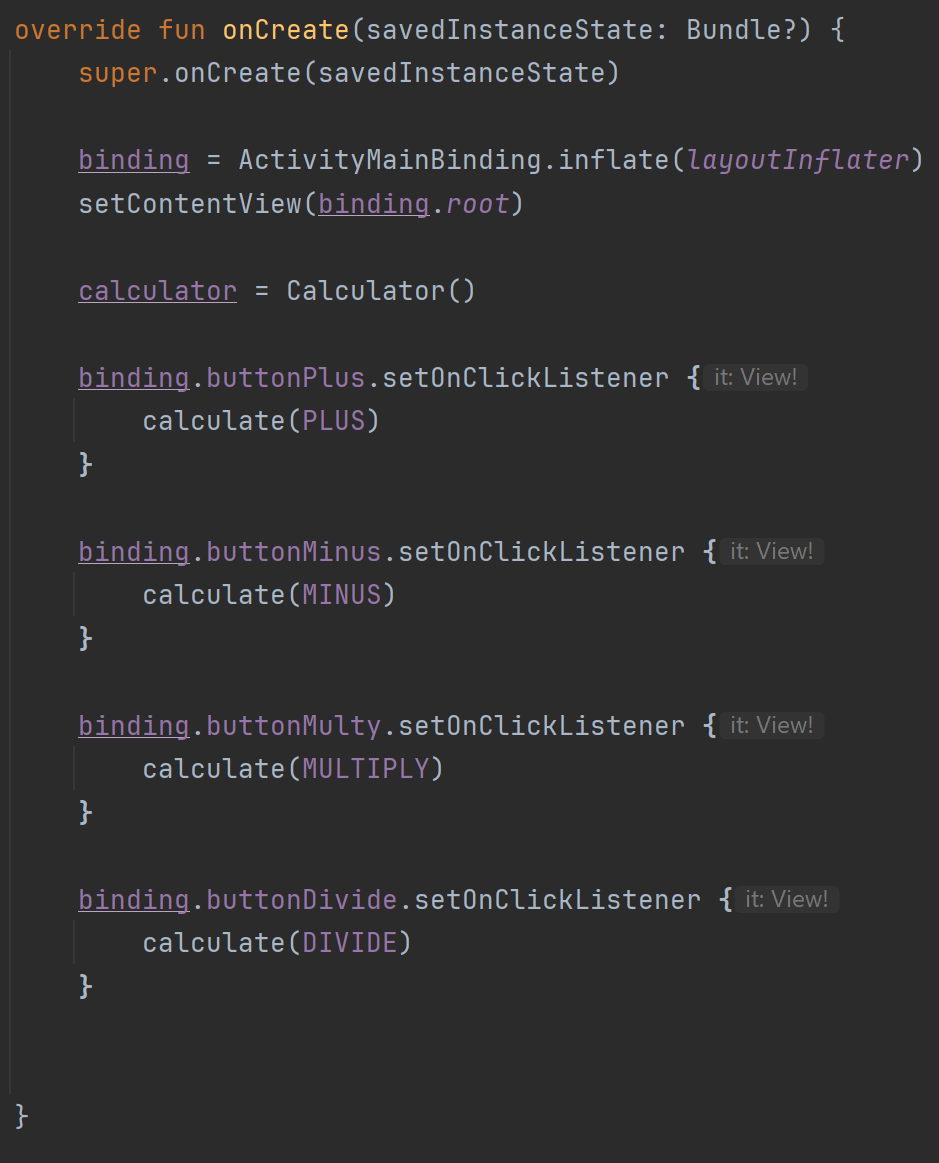


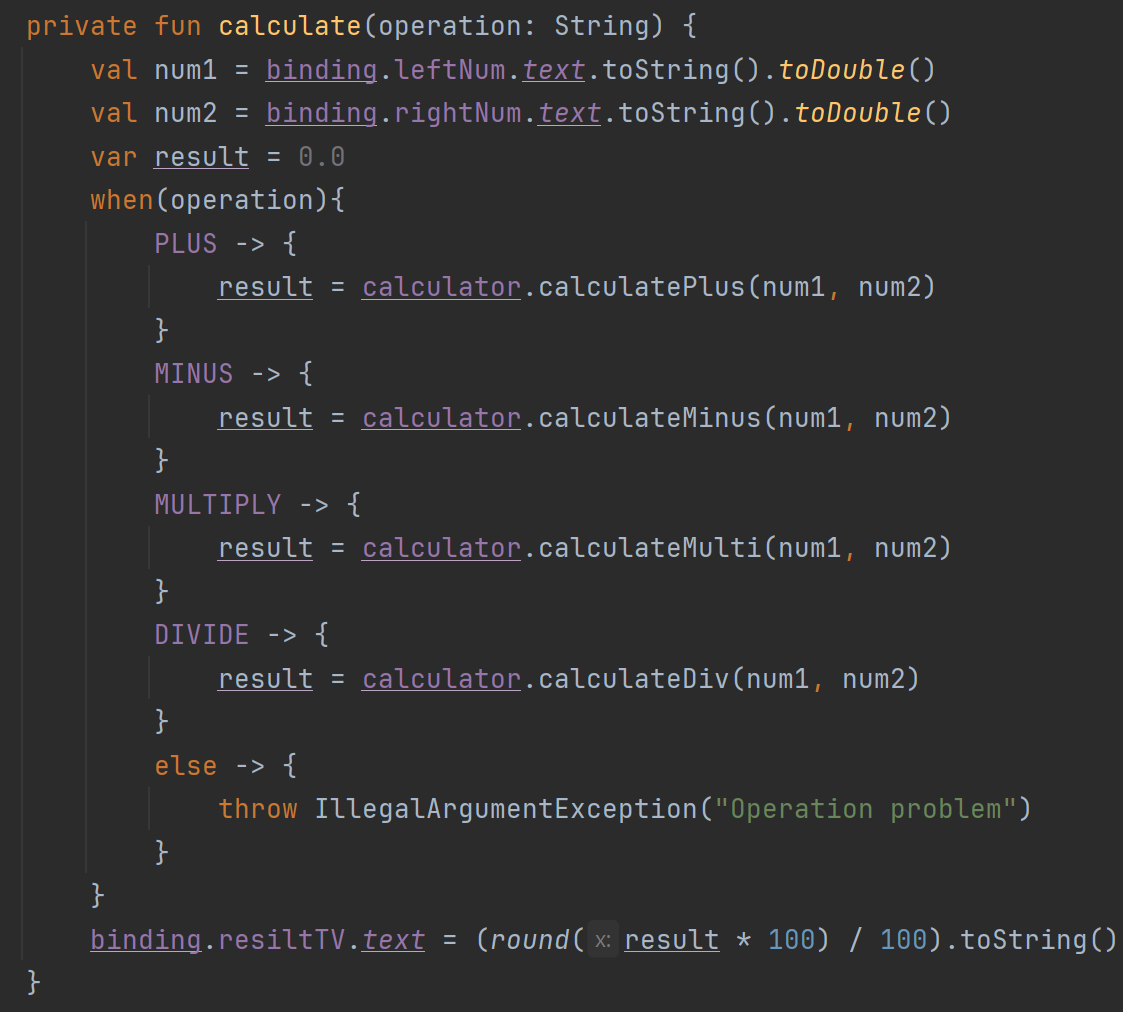


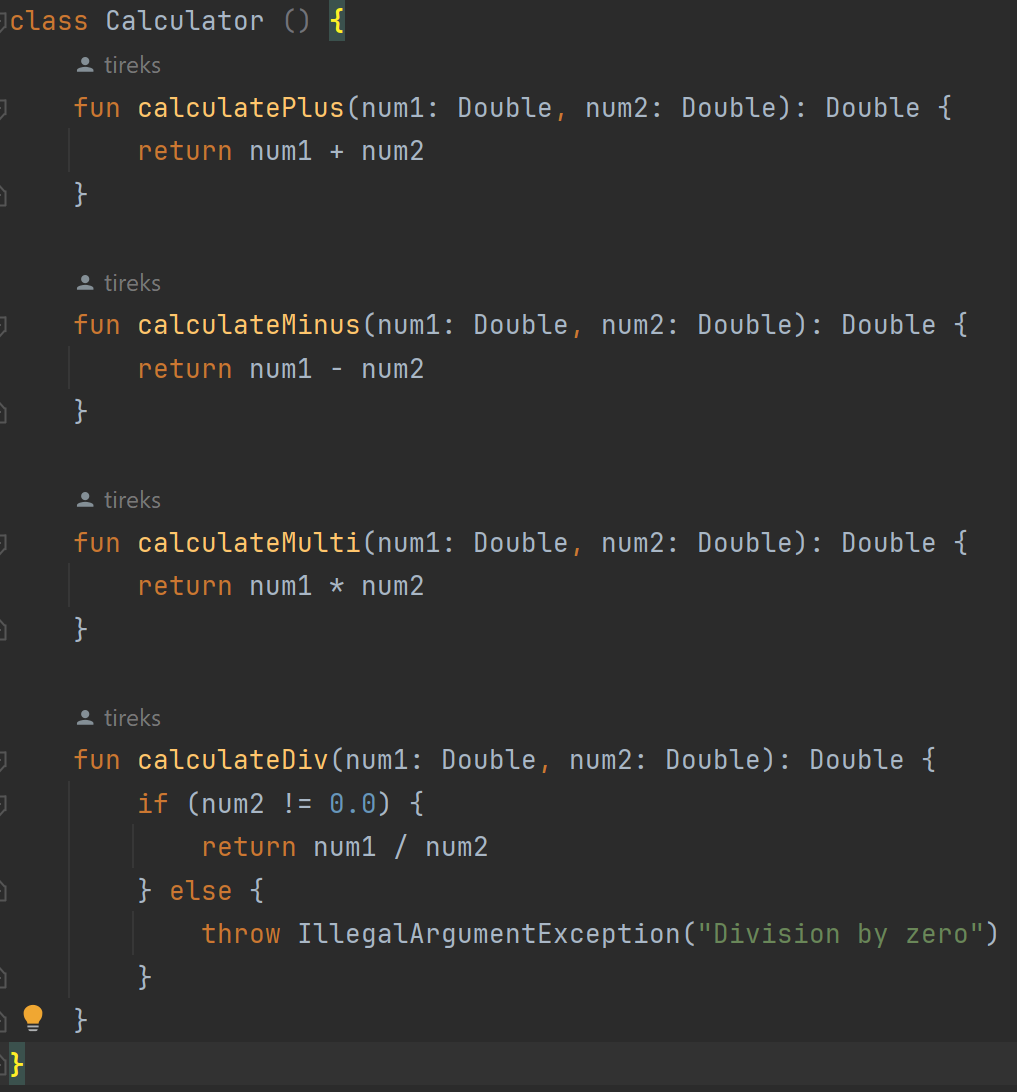


Приложение без применения паттерна:

Это также калькулятор, но без использования паттерна Стратегия. В этой версии приложения все операции выполняются непосредственно в методе обработчика событий, а выбор операции осуществляется путем передачи символа операции (например, +, -, \*, /) в метод вычисления.







Обоснование использования паттерна Стратегия:

Использование паттерна Стратегия обусловлено необходимостью создания гибкой и расширяемой системы, в которой операции могут легко добавляться и изменяться без модификации основного кода. Паттерн Стратегия позволяет выделить различные операции в отдельные классы-стратегии, обеспечивая легкость в управлении и поддержке кода приложения.

Сравнительный анализ:

Преимущества использования паттерна Стратегия:

* Гибкость и расширяемость: новые операции могут быть легко добавлены, не затрагивая существующий код.
* Читаемость и понятность кода: разделение логики на отдельные классы-стратегии делает код более организованным и легким для понимания.
* Повторное использование кода: благодаря отделению алгоритмов от контекста, стратегии могут быть повторно использованы в разных контекстах.

Недостатки использования паттерна Стратегия:

* Дополнительная сложность: добавление дополнительных классов-стратегий может увеличить сложность приложения.
* Необходимость внедрения: не всегда целесообразно использовать паттерн Стратегия для решения простых задач.

Выводы:

Паттерн Стратегия является мощным инструментом для обеспечения гибкости и расширяемости приложений. Его использование обеспечивает лучшую модульность и читаемость кода, что способствует разработке более надежных и поддерживаемых приложений. Однако, в некоторых случаях, особенно при решении простых задач, использование этого паттерна может быть излишним и ведет к избыточной сложности кода.