Chain Response Program - Design Document
by 21800436_HeechanYang

Introduction

This program uses Chain of Responsibility structure to solve a simple math equation with equation limited to addition, subtraction, multiplication and division. Program chains by the order of add -> sub -> mult -> div. This means that the input of two integers and one string expressing the operand will be sent to in this order.

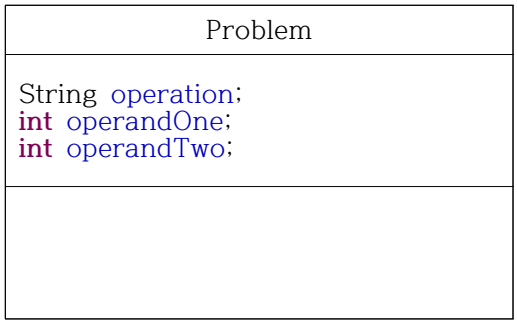
Class Diagram

Main
public static void main(String[] args)

1. Main
    a. Field : none
    b. Method :
        public static void main(String[] args)
            1. Uses Start class to run the program.
            2. By do{}while() operation, the program is looped until 'term' is typed into operand option.

Start
private int count = 1; private Problem problem;
public void start() private Problem inputProblem()

2. Start
    a. Field :
        count : is an integer type variable that keeps count of how many calculation ran.
        problem : is a Problem type variable for the input of use choses operand and operations.
    b. Method :
        Start() : makes Solve type variable for each operations and 'Chain' these variables in given order. These are each server made for each equation. The first problem is sent to 'addition' server.
        inputProblem() : receives users input of an operation and two operands and returns it as Problem type. And when the operation is 'term' the program is terminated.



3. Problem

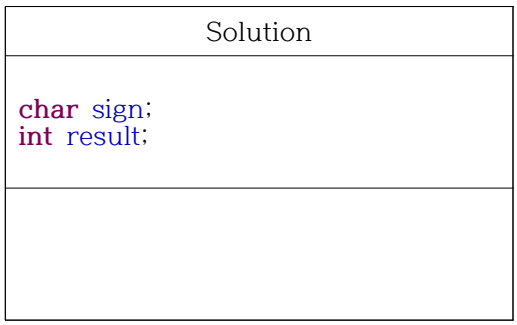
Field :

operation : string type variable that has users input of which operation to run.

operandOne : integer type variable that has users input of the first integer for calculation.

operandTwo : integer type variable that has users input of the Second integer for calculation.

Method : none



4. Problem

Field :

sign : saves the sign of operation for result output.

result : saves the integer type result from the operation for output.

Method : none

Solve
private String calcName; private Solve next; private Solution solution;
public Solve(String calcName) public Solve setNext(Solve next) public final void solve(Problem problem) protected abstract boolean check(Problem problem) protected abstract Solution resolve(Problem problem) protected void done(Problem problem, Solution solution) protected void fail(Problem problem)

5. Solve

Field :

calcName: a string type variable that receives the word of operation when each Solve variable are made.

next : indicates the next server.

solution : receives the answer and the information to be printed as an output.

Method :

Solve(String CalcName) : saves the string type word of operation.

setNext(Solve next) : sets the next server.

solve(Problem problem) : function that determines the operations given.

check(Problem problem) : checks whether the user inputted operation can be solved in current server.

resolve(Problem problem) : abstract method that is implement differently in each server. (will be explained in each class servers)

done(Problem problem, Solution solution) : ran when the problem is solved, shows what equation it ran and solved.

fail(Problem problem) : when the user inputs an operation that is not included in the program it shows that there are no such operation.

Addition
private Solution solution;
public Addition(String calcName) protected boolean check(Problem problem) protected Solution resolve(Problem problem)

6. Addition : class that extends Solve class.

Field :

solution : Solution type variable which saves the result from the operation.

Method :

Addition : calls super class and saves the name of the operation.

check : checks whether the input operation should be solved in current server. Return boolean if the operation is in the right server.

resolve : return the solution with implementation for the operation.

Subtraction
private Solution solution;
public Subtraction(String calcName) **protected boolean** check(Problem problem) **protected** Solution resolve(Problem problem)

7. Subtraction: class that extends Solve class.
    Field :
        solution : Solution type variable which saves the result from the operation.
    Method :
        Subtraction : calls super class and saves the name of the operation.
        check : checks whether the input operation should be solved in current server. Return boolean if the operation is in the right server.
        resolve : return the solution with implementation for the operation.

Multiplication
private Solution solution;
public Multiplication(String calcName) **protected boolean** check(Problem problem) **protected** Solution resolve(Problem problem)

8. Subtraction: class that extends Solve class.
    Field :
        solution : Solution type variable which saves the result from the operation.
    Method :
        Multiplication : calls super class and saves the name of the operation.
        check : checks whether the input operation should be solved in current server. Return boolean if the operation is in the right server.
        resolve : return the solution with implementation for the operation.

Division
private Solution solution;
public Division(String calcName) **protected boolean** check(Problem problem) **protected** Solution resolve(Problem problem)

9. Subtraction: class that extends Solve class.
    Field :
        solution : Solution type variable which saves the result from the operation.
    Method :
        Division : calls super class and saves the name of the operation.
        check : checks whether the input operation should be solved in current server. Return boolean if the operation is in the right server.
        resolve : return the solution with implementation for the operation