ㅣ

**KIM, DAHYE (김다혜)**

# github URL (optional):

|  |  |
| --- | --- |
|  | [과제의 목적을 간단하게 설명] |

# 개요

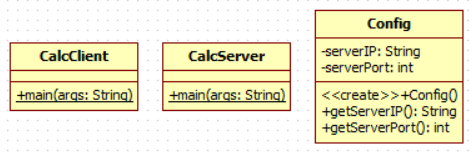
|  |  |
| --- | --- |
|  | **JAVA socket API를 사용해 client-server 구조의 internet 계산기 프로그램 제작** |

**Requirements**

* **Client에서 사용자로부터 사칙연산 수식을 입력받은 후 그 수식을 Server로 보냄. Server는 수식을 해석한 후 계산하고, 계산 결과를 Client에게 보냄. 계산 결과를 받은 Client가 계산 결과를 출력하는 계산기 프로그램 제작.**
* **Client와 Server 간의 communication은 아스키코드 기반의 message format을 가진 직접 정의한 application protocol로 이루어지도록 프로그램 제작.**
* **계산기는 +, - , \*, /의 기본적인 사칙연산을 수행할 수 있어야 함.**
* **Server가 계산한 결과는 수식에 대한 답 또는 error message 일 수 있음.**
* **Server는 동시에 여러 개의 Client를 다룰 수 있는 mult thread 구조를 가져야 함.**
* **Client가 Server의 IP주소, Server의 port number를 file로부터 읽어오도록 하고, file이 존재하지 않는 경우 default 정보를 지정하여 처리하도록 함.**

# REPORT

* **구조도**



# PRotocol

|  |  |
| --- | --- |
|  | **Client와 Server 간의 request와 response를 위한 아스키코드 기반의 message format을 가진 application protocol을 직접 정의.** |

* **Request message format : (연산 type) (피연산자1) (피연산자2)**

|  |  |
| --- | --- |
| **사용자가 입력한 수식** | **Protocol message** |
| **20 + 10** | **ADD 20 10** |
| **20 - 10** | **SUB 20 10** |
| **20 \* 10** | **MUL 20 10** |
| **20 / 10** | **DIV 20 10** |
| **20 (지원하지 않는 연산자) 10** | **OPER 20 10** |
| **인자가 3개가 아닌 경우** | **ARG** |

* **Response message format: (정답type) (계산 결과) or (에러 type) (에러 의미)**

|  |  |  |
| --- | --- | --- |
| **수식** | **Protocol message** | **Client 출력 결과** |
| **20 + 10 = 30** | **ANS 30** | **Answer: 30** |
| **20 / 0** | **ERRDIV Division error** | **Error message: divided by zero** |
| **20 (지원하지 않는 연산자) 10** | **ERROPER Operation error** | **Error message: invalid operation** |
| **인자가 3개가 아닌 경우** | **ERRARG Argument error** | **Error message: too many arguments** |

SOURCE CODES

## **Config.java**

|  |
| --- |
| importjava.io.\*;  importjava.util.\*;  public class Config {  private String serverIP;  private int serverPort;  public Config() {  //Default Settings Information  serverIP="localhost";  serverPort=1234;  try {  //Read server IP address and server port number from file  Properties properties=new Properties();  properties.load(new FileInputStream("src/server\_info.dat"));  if(properties.containsKey("serverIP")) {  serverIP=properties.getProperty("serverIP");  }  if(properties.containsKey("serverPort")) {  serverPort=Integer.*parseInt*(properties.getProperty("serverPort"));  }  }catch(IOException e) {  System.***out***.println(e.getMessage());  }  }  public String getServerIP() { //Returns server IP address  return serverIP;  }  public int getServerPort() { //Returns server port number  return serverPort;  }  } |

## **CalcServer.java**

|  |
| --- |
| importjava.io.\*;  importjava.net.\*;  importjava.util.concurrent.\*;  public class CalcServer {  public static void main(String[] args) {  // **TODO**Auto-generated method stub  Config config=new Config();  int port=config.getServerPort(); //Port number setting  try {  ServerSocket listener=new ServerSocket(port); //Create server socket  ExecutorService pool=Executors.*newFixedThreadPool*(20); //Create a thread pool that can process up to 20 tasks simultaneously => multithread  while(true) {  Socket sock=listener.accept(); //Waiting for connection request from client  pool.execute(new Calc(sock)); //Executes run() of Calcclass in thread  }  }catch(IOException e) {  System.***out***.println(e.getMessage());  }  }  private static class Calc implements Runnable{  private Socket socket;  Calc(Socket socket){  this.socket=socket;  }  public void run() {  try {  //Create input/output stream  BufferedReader in=new BufferedReader(new InputStreamReader(socket.getInputStream()));  BufferedWriter out=new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));  while(true) {  String inputMessage=in.readLine(); //Receive data from client  if(inputMessage.equalsIgnoreCase("bye")) {  break;  }  String[] parts=inputMessage.split(" ");  //Calculate with data  if(parts.length!=3) {  out.write("ERRARG Argument error"+"\n"); //Send data to client  out.flush();  }  else {  String operation=parts[0];  int operand1=Integer.*parseInt*(parts[1]);  int operand2=Integer.*parseInt*(parts[2]);  int result;  String outputMessage;  switch(operation) {  case "ADD":  result=operand1+operand2;  outputMessage="ANS "+result;  break;  case "SUB":  result=operand1-operand2;  outputMessage="ANS "+result;  break;  case "MUL":  result=operand1\*operand2;  outputMessage="ANS "+result;  break;  case "DIV":  if(operand2==0) {  outputMessage="ERRDIV Division error";  }  else {  result=operand1/operand2;  outputMessage="ANS "+result;  }  break;  default:  outputMessage="ERROPER Operation error";  }  out.write(outputMessage+"\n"); //Send data to client  out.flush();  }  }  }catch(IOException e) {  System.***out***.println(e.getMessage());  }finally {  try {  if(socket!=null) {  socket.close(); //Close socket  }  }catch(IOException e) {  System.***out***.println(e.getMessage());  }  }  }  }  } |

## Cal**cClient.java**

|  |
| --- |
| importjava.io.\*;  importjava.net.\*;  importjava.util.\*;  public class CalcClient {  public static void main(String[] args) {  // **TODO**Auto-generated method stub  Config config=new Config();  String serverIP=config.getServerIP(); //Server IP address settings  int serverPort=config.getServerPort(); //Server port number settings  try {  Socket socket=new Socket(serverIP, serverPort); //Request to connect to server  //Create input/output stream  BufferedReader in=new BufferedReader(new InputStreamReader(socket.getInputStream()));  BufferedWriter out=new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));  Scanner scanner=new Scanner(System.***in***);  while(true) {  System.***out***.println("Please enter the calculation formula (enter with spaces, ex) 10 + 20 ): ");  String inputMessage=scanner.nextLine(); //Receive formula from the user  if(inputMessage.equalsIgnoreCase("bye")) {  out.write(inputMessage); //Send data to server  out.flush();  break;  }  String[] parts1=inputMessage.split(" ");  //Converting an expression to protocol message  if(parts1.length!=3) {  out.write("ARG"+"\n"); //Send data to server  out.flush();  }  else {  String operand1=parts1[0];  String operator=parts1[1];  String operand2=parts1[2];  String outputMessage;  switch(operator) {  case "+":  outputMessage="ADD "+operand1+" "+operand2;  break;  case "-":  outputMessage="SUB "+operand1+" "+operand2;  break;  case "\*":  outputMessage="MUL "+operand1+" "+operand2;  break;  case "/":  outputMessage="DIV "+operand1+" "+operand2;  break;  default:  outputMessage="OPER "+operand1+" "+operand2;  }  out.write(outputMessage+"\n"); //Send data to server  out.flush();  }  String resultMessage=in.readLine(); //Receive data from server  String[] parts2=resultMessage.split(" ");  //Converting protocol messages to calculated results  String para1=parts2[0];  String para2=parts2[1];  String result;  switch(para1) {  case "ANS":  result="Answer: "+para2;  break;  case "ERRARG":  result="Error message: too many arguments";  break;  case "ERRDIV":  result="Error message: divided by zero";  break;  case "ERROPER":  result="Error message: invalid operation";  break;  default:  result="Error message";  }  System.***out***.println(result); //Outputting Calculation Results  }  }catch(IOException e) {  System.***out***.println(e.getMessage());  }  }  } |

## OUTPUT (Screen Shots)

|  |  |
| --- | --- |
| #1 | **계산기 프로그램 실행 결과** |
|  | |