**package** io;

**import** java.awt.Font;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.awt.event.WindowAdapter;

**import** java.awt.event.WindowEvent;

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.File;

**import** java.io.FileReader;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** javax.swing.JFileChooser;

**import** javax.swing.JFrame;

**import** javax.swing.JOptionPane;

**import** javax.swing.JScrollPane;

**import** javax.swing.JTextArea;

**import** javax.swing.ScrollPaneConstants;

**public** **class** TryNotepad **extends** JFrame **implements** ActionListener {

**private** JTextArea area; // 글씨 쓰는 곳

**private** MenuPane menu; // Component인 menu를 Frame이 있는 곳에 붙여넣기

**private** File file = **null**;

**public** TryNotepad() {

area = **new** JTextArea();

area.setFont(**new** Font("고딕체", Font.***BOLD***, 20)); // 폰트 다르게 굵게 20으로

JScrollPane scroll = **new** JScrollPane(area); // area에 scroll 넣어주기

scroll.setVerticalScrollBarPolicy(ScrollPaneConstants.***VERTICAL\_SCROLLBAR\_ALWAYS***);

menu = **new** MenuPane();

**this**.setJMenuBar(menu); // JFrame(this)= setJMenuBar, MenuBar에다 menu를 넣겠다는 의미

**this**.add("Center", scroll); // 굳이 add를 쓰지 않아도 setJMenuBar에 의해 꼭대기에 menu를 붙여준다

setBounds(900, 100, 500, 500);

setVisible(**true**);

// setDefaultCloseOperation(EXIT\_ON\_CLOSE);

**this**.setDefaultCloseOperation(***DO\_NOTHING\_ON\_CLOSE***);

// 여기서 this는 JOPtionPane이 아니라 JFrame에 붙어 있는 거

// 기본적으로 JFrame은 숨는 게 기본 특징이다!!! 아무말 안하면 hide를 기본으로 갖는다

// DISPOSE는 사후 처리.

// 취소를 눌렀을 때 창이 그대로 살아있게 하기 위함으로 쓰는거당

**this**.addWindowListener(**new** WindowAdapter() {

// 여기서 this는 생략해도 괜찮다 Frame을 말한다. 구현부를 new WindowAdapter가 쥐고 있다

@Override

**public** **void** windowClosing(WindowEvent e) {

**int** result = JOptionPane.*showConfirmDialog*(TryNotepad.**this**, // 여기서 말하는 this는 익명이너를 말한다.

"저장하시겠습니까?", "메모장", JOptionPane.***YES\_NO\_CANCEL\_OPTION***, JOptionPane.***WARNING\_MESSAGE***);

**if** (result == JOptionPane.***YES\_OPTION***) {

System.*exit*(0);

} **else** **if** (result == JOptionPane.***NO\_OPTION***) {

saveDialog();

filewrite();

System.*exit*(0);

}

}

});

// 이벤트 처리 - 메뉴

menu.getNewM().addActionListener(**this**); // 구현부는 나(this)가 갖고 있다

// 앞에 this.newM.addAction~~~ 이런식으로 써주면 내 클래스 안에서 newM을 찾으라는 의미인데,

// 눈을 씻고 찾아봐도 내 클래스 안에는 newM이 없다

// 또한 getnewM()이 아닌 그냥 newM이라고 쓰게 되면 newM이 private으로 잡혀 있기 때문에 다른 클래스에서 쓸 수 없다

// 접근이 안되니까 뭘 잡아야 할까? 값을 꺼내갈 수 있도록 getter를 잡아줘야 한다!

menu.getOpenM().addActionListener(**this**);

menu.getSaveM().addActionListener(**this**);

menu.getExitM().addActionListener(**this**);

menu.getCutM().addActionListener(**this**);

menu.getCopyM().addActionListener(**this**);

menu.getPasteM().addActionListener(**this**);

}

@Override

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == menu.getNewM()) {

area.setText(""); // 초기화

} **else** **if** (e.getSource() == menu.getOpenM()) {

openDialog(); // 열기 다이얼로그 -> 파일 선택

fileRead();// -> 파일의 내용을 읽어서 JFrame의 TextArea에 뿌리기

} **else** **if** (e.getSource() == menu.getSaveM()) {

JFileChooser chooser = **new** JFileChooser(); // 원하는 저장 위치 선택

**int** result = chooser.showSaveDialog(**this**); // 저장 다이얼로그

File file = **null**;

**if** (result == JFileChooser.***APPROVE\_OPTION***) { // save버튼 눌렀으면

file = chooser.getSelectedFile(); // 선택한 파일을 가져와라

**try** {

BufferedWriter bw = **new** BufferedWriter(**new** FileWriter(file));

bw.close();

} **catch** (IOException e1) {

e1.printStackTrace();

}

System.*exit*(0);

}

} **else** **if** (e.getSource() == menu.getExitM()) {

// 다이얼로그

**int** result = JOptionPane.*showConfirmDialog*(TryNotepad.**this**, // JFrame 위에다 띄어주기

"저장하시겠습니까?", // 메시지 띄우기

"메모장", // 타이틀

JOptionPane.***YES\_NO\_CANCEL\_OPTION***, // 예 아니오 취소 옵션

JOptionPane.***QUESTION\_MESSAGE***); // ? 물음표 .... ! - Warning, i - information

// \*\*\*yes든 no든 전부 int형(정수형)으로 인식한다

// YES\_NO\_CANCEL\_OPTION가 대문자로 되어 있다는 건 상수로 되어 있어 숫자로 인식한다

// 예스든 노든 취소든 누르면 나한테 int형으로 넘어온다

**if** (result == 0) { // YES\_OPTION

saveDialog();// 파일을 저장하고~

filewrite();

System.*exit*(0); // 종료

} **else** **if** (result == JOptionPane.***NO\_OPTION***) { // 이렇게 쓰는 게 더 보기 편하니 더 좋다!

System.*exit*(0);

} **else** **if** (result == JOptionPane.***CANCEL\_OPTION***) {

}

} **else** **if** (e.getSource() == menu.getCutM()) {

area.cut();

} **else** **if** (e.getSource() == menu.getCopyM()) {

area.copy();

} **else** **if** (e.getSource() == menu.getPasteM()) {

area.paste();

}

} // actionPerformed(Actionevent e)

**private** **void** filewrite() {

**if** (file == **null**)

**return**;

String data = **null**; // getText니까 String type으로

**try** {

BufferedWriter bw = **new** BufferedWriter(**new** FileWriter(file));

area.getText(); // 값을 몽땅 꺼내와라

bw.write(data); // 파일로 저장

bw.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

**private** **void** saveDialog() {

JFileChooser chooser = **new** JFileChooser(); // 원하는 저장 위치 선택

**int** result = chooser.showSaveDialog(**null**); // 저장 다이얼로그

File file = **null**;

**if** (result == JFileChooser.***APPROVE\_OPTION***) { // save버튼 눌렀으면

file = chooser.getSelectedFile(); // 선택한 파일을 가져와라

}

JOptionPane.*showMessageDialog*(**this**, file);

}

**private** **void** fileRead() {

**if** (file == **null**)

**return**;

area.setText("");

**try** {

BufferedReader br = **new** BufferedReader(**new** FileReader(file));

// 열기에서 파일을 넣지 않고 취소 버튼을 눌렀을 때 NullPointerException 오류가 뜬다

// file이 Null이라는 거

String line;

**while** ((line = br.readLine()) != **null**) { // 1줄씩 (Enter를 칠 때까지) 하지만 Enter는 읽지 않는다. Enter전까지.

// area.setText(line); - 읽어온 줄을 setText area에 넣어준다,

// 근데... setText는 덮어쓰기 역할을 해서 파일을 열었는데 글들이 덮어쓰여서 안보이는 것처럼 보인다

area.append(line + "\n"); // readLing이 엔터값을 못 잡으니

// 글이 한줄로 나온다

// 강제로 "\n"을 넣어준다

} // while

// 파일을 한꺼번에 읽어올 수 없다

// 한줄씩 읽어와야 하는데 다 읽으면 null값으로 처리한다

// 한번 읽고 끝나는 게 아니니까 null이 아닌 동안까지 while문을 돌린다

br.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

// 필드에 private File file = null; 넣어주는 방법 대신

// void 메소드에 return값을 넣어주는 것도 방법

**private** **void** openDialog() {

JFileChooser chooser = **new** JFileChooser();

**int** result = chooser.showOpenDialog(**this**);

**if** (result == JFileChooser.***APPROVE\_OPTION***) { // 열기를 선택했다는 거냐고 물어보는 거

file = chooser.getSelectedFile();

}

JOptionPane.*showMessageDialog*(**this**, file);

// file은 클@16진수인데 내가 선택한 파일이 제대로 나왔는지가 궁금한거니까

// tostring으로 알아서 바꿔버리는 거

}

**public** **static** **void** main(String[] args) {

**new** TryNotepad();

}

}

// JFileChooser - 파일 추적

// showOpenDialog - 다이얼로그 열기

// showOpenDialog(Component parent) Component 라는 건 어디에 띄울거냐는 말이다

// showOpenDialog static이 아니다. 그럼 먼저 클래스(JFileChooser)를 생성해 줘야 한다.

// JFileChooser chooser = new JFileChooser();

//showSaveDialog() - 다이얼로그 저장