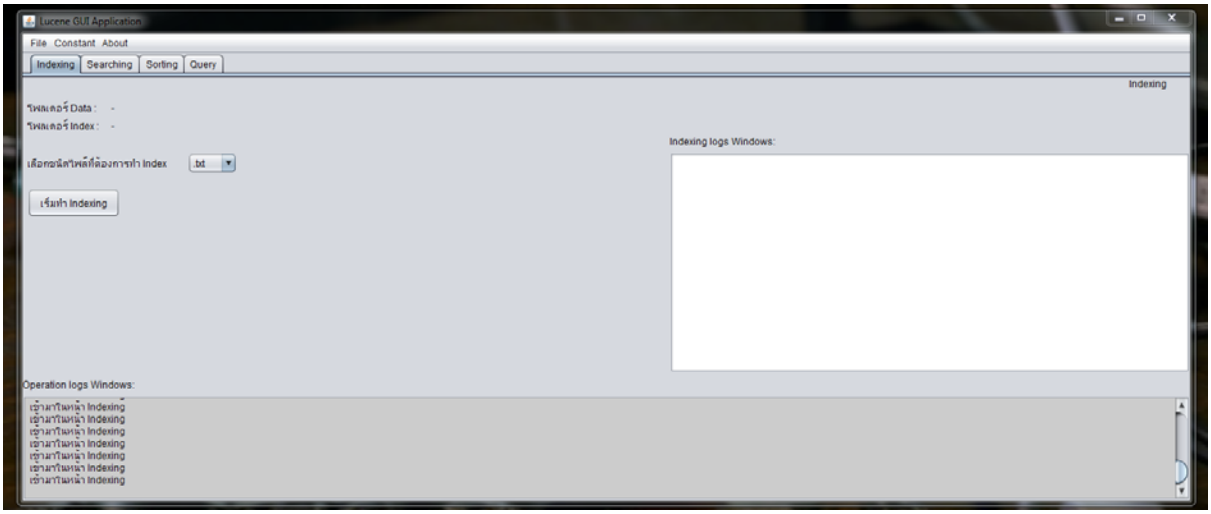
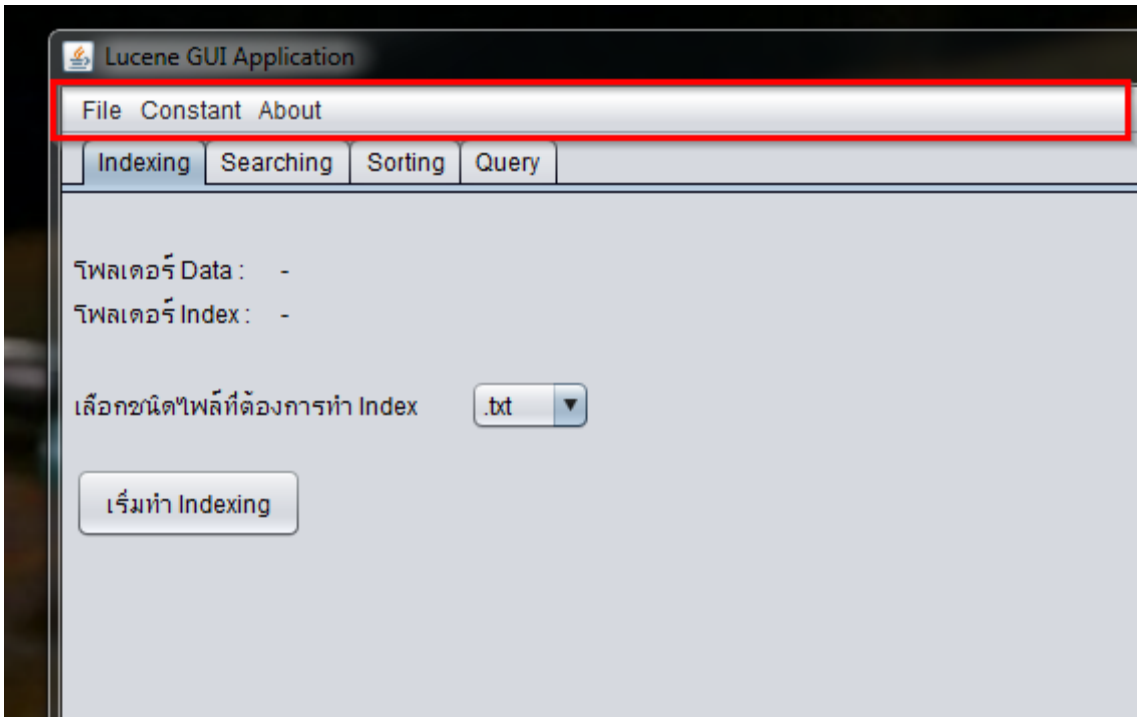
**เอกสารอธิบายการทำงานของโปแกรม**

เมื่อมีการรันโปรแกรมก็จะพบกับหน้าแรกแบบนี้



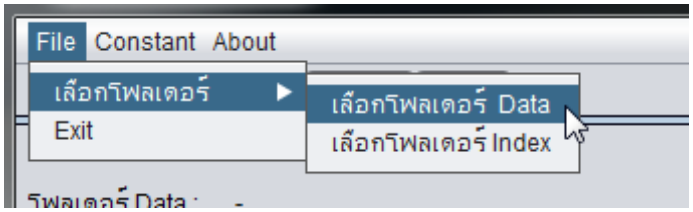

โดยตัวโปรแกรมจะประกอบไปด้วยส่วนต่างๆดังนี้

1.Menu Bar




จะประกอบไปด้วย 3 เมนูคือ เมนู File, Constant, About

ใน Menu File จะประกอบไปด้วย Menu ย่อย การเลือกไฟล์ ซึ่งเป็นส่วนที่ให้ผู้ใช้เลือก path ของโฟลเดอร์ที่เก็บ Data และ Folder ที่ใช้เก็บค่าการทำ Index



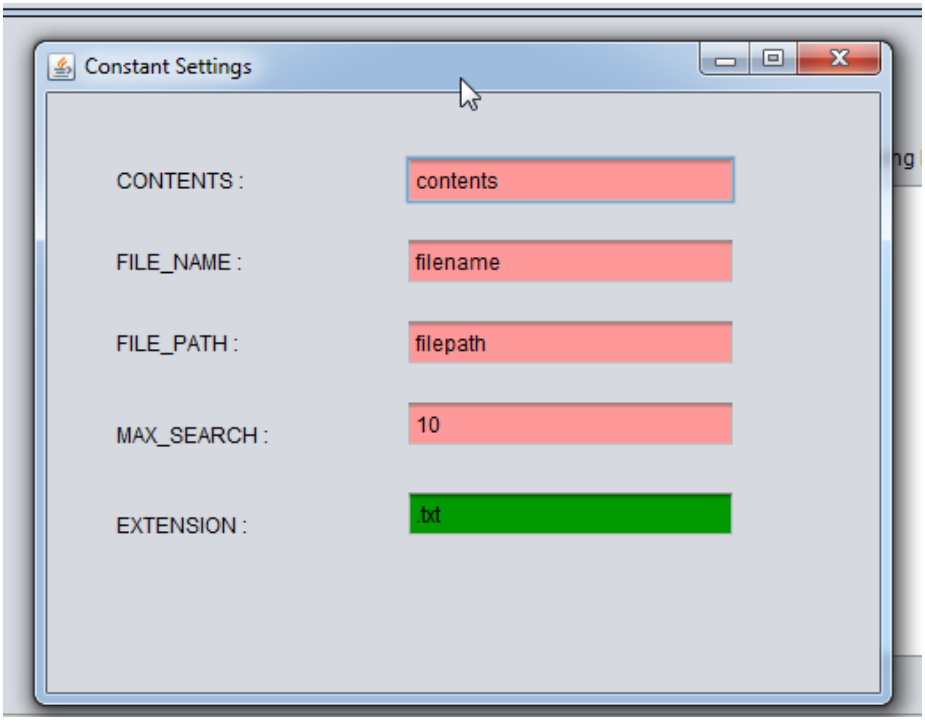

*โค้ดการทำงานในส่วนการเลือกโฟลเดอร์ Data:*

```
    private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:

    jFileChooser1.setCurrentDirectory(new java.io.File("."));
    jFileChooser1.setDialogTitle("เลือกโฟลเดอร์ Data");
    jFileChooser1.setFileSelectionMode(jFileChooser1.DIRECTORIES_ONLY);
    jFileChooser1.setAcceptAllFileFilterUsed(false);

    if (jFileChooser1.showOpenDialog(null) == jFileChooser1.APPROVE_OPTION) {
     // System.out.println("getCurrentDirectory(): " + jFileChooser1.getCurrentDirectory());
     // System.out.println("getSelectedFile() : " + jFileChooser1.getSelectedFile());
      dataPath = ""+jFileChooser1.getSelectedFile();
     // System.out.println(dataPath.replace("\\", "\\\\"));
      dataPath.replace("\\", "\\\\");
      jTextArea1.setText(jTextArea1.getText() + "เลือกโฟลเดอร์ Data: " + dataPath +"\n");
      jLabel7.setText(dataPath);
    } else {
     // System.out.println("No Selection ");
      jTextArea1.setText(jTextArea1.getText() + "ยังไม่ได้เลือกโฟลเดอร์ Data" +"\n");
      jLabel7.setText("ยังไม่ได้เลือกโฟลเดอร์ Data");
    }
    }
```

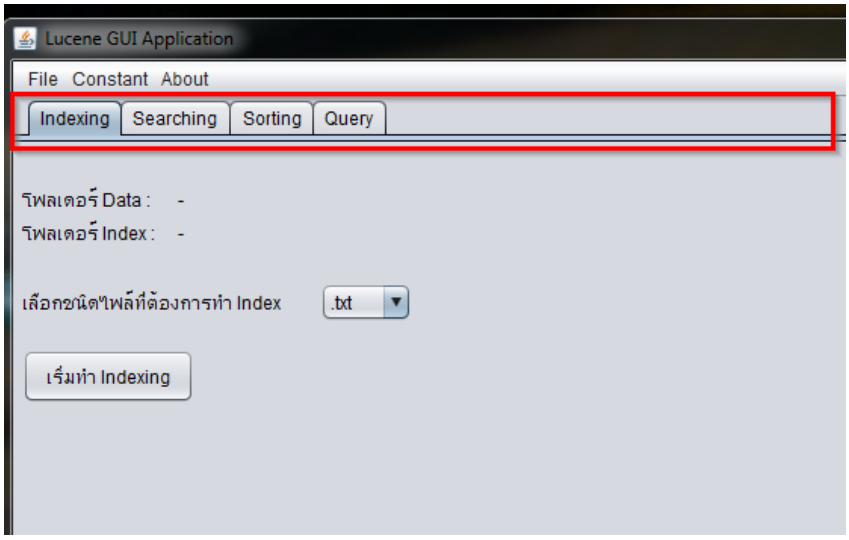
*โค้ดการทำงานในส่วนการเลือกโฟลเดอร์ Index:*

```
    private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt) {
    jFileChooser1.setCurrentDirectory(new java.io.File("."));
    jFileChooser1.setDialogTitle("เลือกโฟลเดอร์ Index");
    jFileChooser1.setFileSelectionMode(jFileChooser1.DIRECTORIES_ONLY);
    jFileChooser1.setAcceptAllFileFilterUsed(false);

    if (jFileChooser1.showOpenDialog(null) == jFileChooser1.APPROVE_OPTION) {
      //System.out.println("getCurrentDirectory(): " + jFileChooser1.getCurrentDirectory());
      //System.out.println("getSelectedFile() : " + jFileChooser1.getSelectedFile());
      indexPath = ""+jFileChooser1.getSelectedFile();

      // System.out.println(dataPath.replace("\\", "\\\\"));
      indexPath.replace("\\", "\\\\");
      jLabel8.setText(indexPath);
      jTextArea1.setText(jTextArea1.getText() + "เลือกโฟลเดอร์ Index: " + indexPath +"\n");

    } else {
      //System.out.println("No Selection ");
      jTextArea1.setText(jTextArea1.getText() + "ยังไม่ได้เลือกโฟลเดอร์ Index" +"\n");
      jLabel8.setText("ยังไม่ได้เลือกโฟลเดอร์ Index");
    }        // TODO add your handling code here:
    }
```

เมนูถัดไปคือ Constant ซึ่งเป็นเมนูที่ใช้ดูค่าคงที่ต่างๆใน Lucene




2.Tab Menu

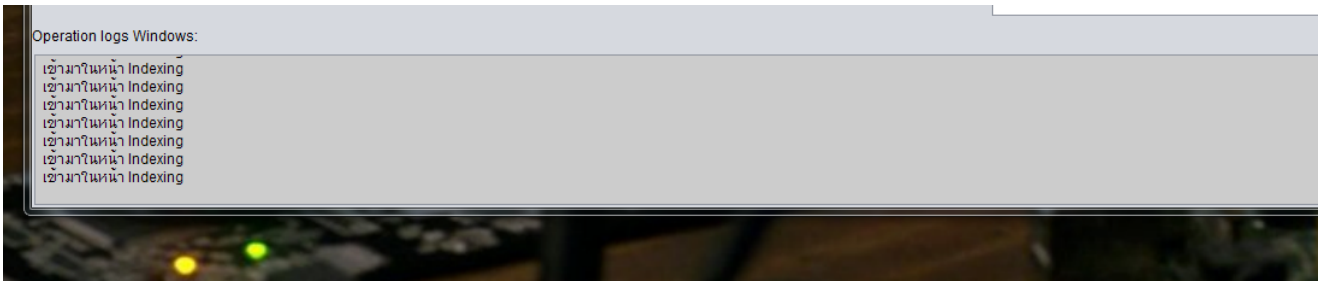



เป็นแท็บที่บอกให้ผู้ใช้เลือกการทำงานในรูปแบบต่างๆของ Lucene ประกอบไปด้วยการทำ Index การทำ Searching การทำ Sorting และการทำ Query

3.Log Windows

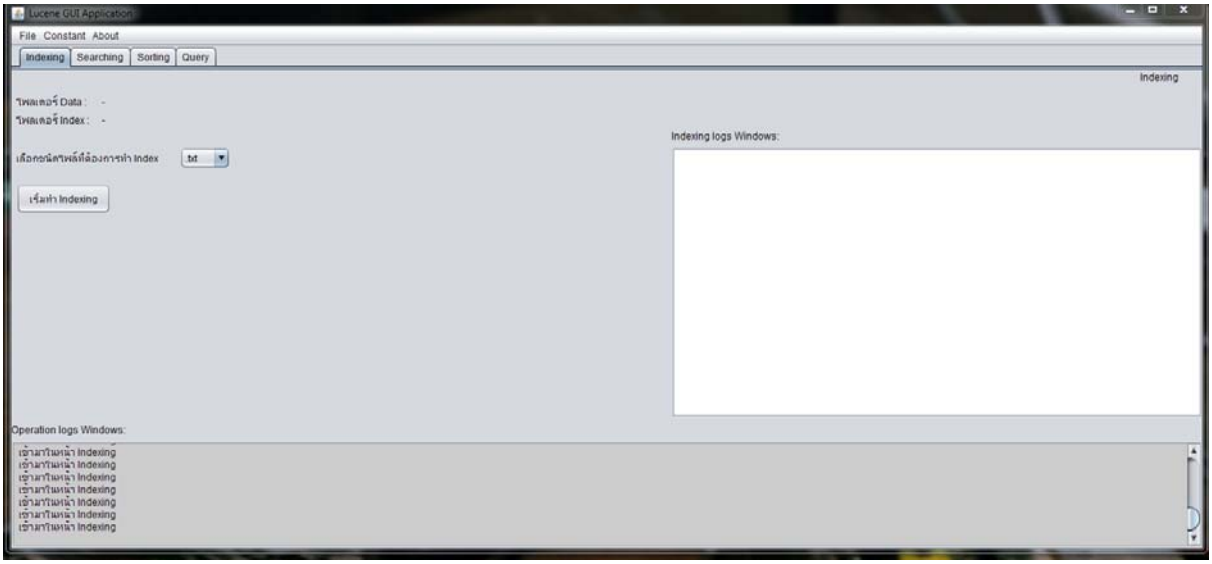
จะไปประกอบไปด้วย 2 แบบคือ

1.Operation Logs Windows (Log การทำงานโดยรวมของโปรแกรม)

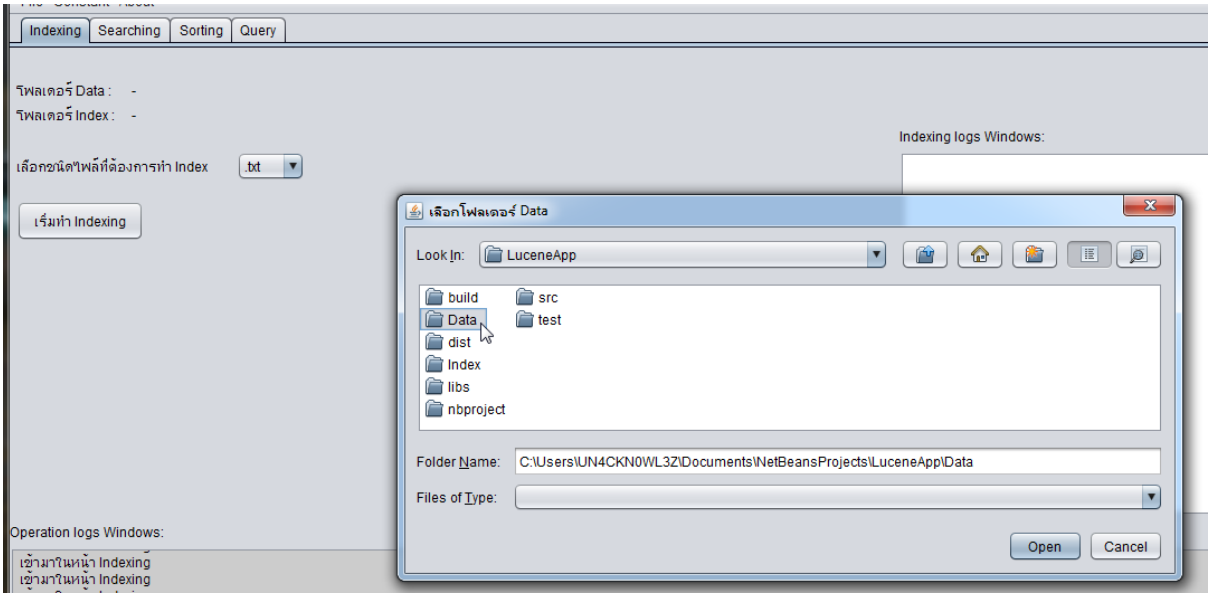



2.Processing Logs Windows คือส่วนที่ใช้แสดงผล ผลลัพธ์การทำงานของกระขวนการทำงานต่างๆ

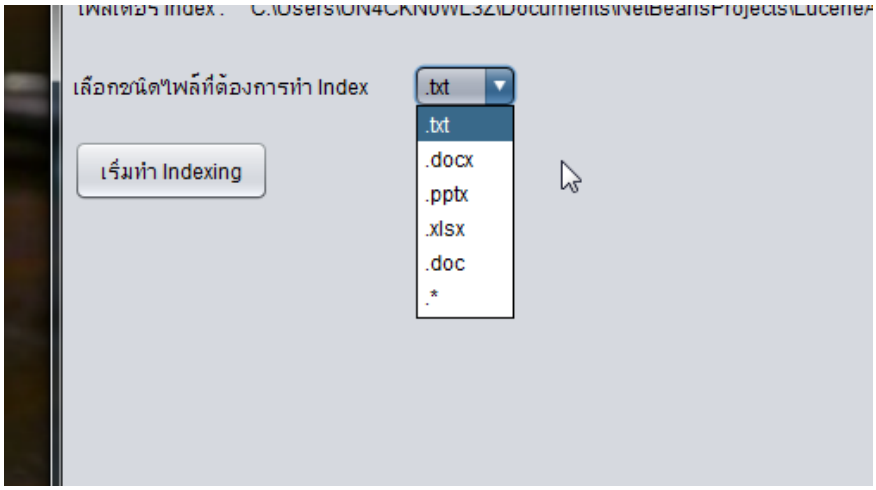
## การทำ Indexing




ขั้นตอนแรกในการทำ Indexing คือผู้ใช้จะต้องเลือกโฟลเดอร์ Data และ Index ก่อนซึ่งเลือกได้ที่เมนูไฟล์



จากนั้นเลือก ชนิดไฟล์ว่าต้องใช้ไฟล์แบบไหน

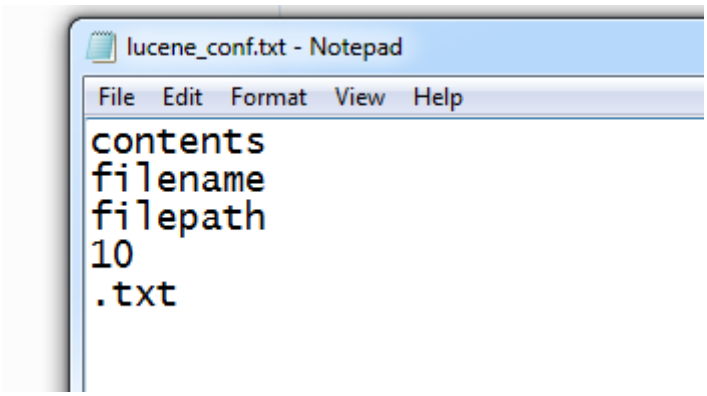



โดยโปแกรมจะมีให้เลือกไฟล์ต่างๆเช่น .txt,docx,pptx,xlsx,doc

*โค้ดการทำงานใน Combobox*

```
private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    NewJFrame1 nj1 = new NewJFrame1();
    if (jComboBox1.getSelectedItem() != null) {
       // System.out.println(jComboBox1.getSelectedItem().toString());
       if(jComboBox1.getSelectedItem().toString().equals(".*")){
           jTextArea1.setText(jTextArea1.getText() + "เลือกชนิดไฟล์ : "+ "เลือกทุกไฟล์" +"\n");
           num_ext = jComboBox1.getSelectedIndex();
           jTextArea1.setText(jTextArea1.getText() + "ComboBox index : "+ num_ext +"\n");
           createConf(nj1.CONTENTS,nj1.FILE_NAME,nj1.FILE_PATH,nj1.MAX_SEARCH,jComboBox1.getSelectedItem().toString());
           jTextArea1.setText(jTextArea1.getText() + "บันทึกค่า extension : "+ jComboBox1.getSelectedItem().toString() +" ลง lucene_conf.txt แล้ว\n");
       }else{
        jTextArea1.setText(jTextArea1.getText() + "เลือกชนิดไฟล์ : "+jComboBox1.getSelectedItem().toString() +"\n");
        num_ext = jComboBox1.getSelectedIndex();
        jTextArea1.setText(jTextArea1.getText() + "เลือก index : "+ num_ext +"\n");
        createConf(nj1.CONTENTS,nj1.FILE_NAME,nj1.FILE_PATH,nj1.MAX_SEARCH,jComboBox1.getSelectedItem().toString());
        jTextArea1.setText(jTextArea1.getText() + "บันทึกค่า extension : "+ jComboBox1.getSelectedItem().toString() +" ลง lucene_conf.txt แล้ว\n");
       }
    }

}
```

แปรแกรมจะบันทึกค่าที่ผู้ใช้เลือกลงในไฟล์ชื่อ lucene_conf.txt ซึ่งจะถูกสร้างในโฟลเดอร์เดียวกันกับโปรแกรม

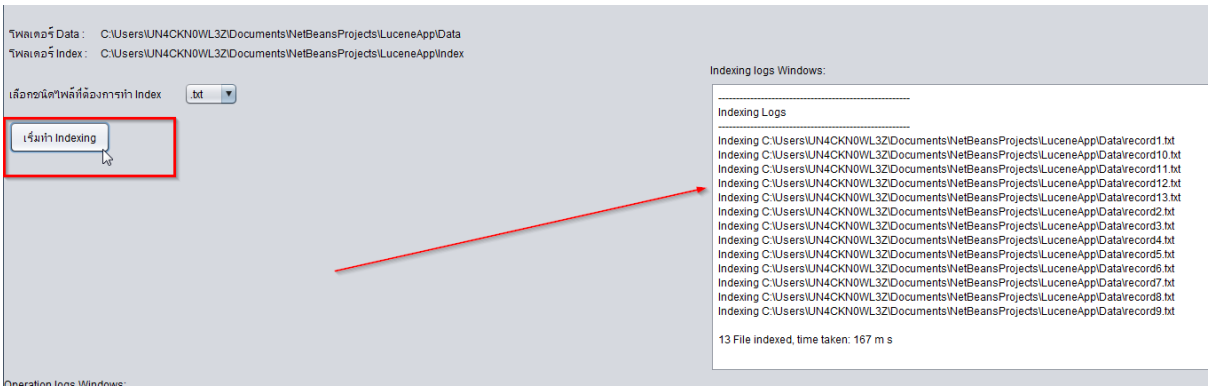


จากนั้นเมื่อมีการทำงานของโปรแกรมคลาส TextFileFilter ก็จะไปอ่านค่าจากไฟล์ lucene_conf.txt อีกทีนึง

```
String CONTENTS;
String FILE_NAME;
String FILE_PATH;
String MAX_SEARCH;
String EXTENSION;
 /**
  * Creates new form NewJFrame1
  */
 public NewJFrame1() {
     initComponents();
     try {
         Scanner sc = new Scanner(new File("lucene_conf.txt"));
         List<String> lines = new ArrayList<String>();
         while (sc.hasNextLine()) {
         lines.add(sc.nextLine());
         }
         String[] arr = lines.toArray(new String[0]);
         jTextField1.setText(arr[0].toString());
         jTextField2.setText(arr[1].toString());
         jTextField3.setText(arr[2].toString());
         jTextField4.setText(arr[3].toString());
         jTextField5.setText(arr[4].toString());

         this.CONTENTS = arr[0].toString();
         this.FILE_NAME = arr[1].toString();
         this.FILE_PATH = arr[2].toString();
         this.MAX_SEARCH = arr[3].toString();
         this.EXTENSION = arr[4].toString();
     } catch (Exception e) {
     }
```

```
NewJFrame1 nj1 = new NewJFrame1();

@ Override


public boolean accept(File pathname) {

        /*
         Pattern p = Pattern.compile("\\.(txt|docx|xlsx|pptx)$");
         String path = pathname.getName().toLowerCase();
         return p.matcher(path).find();
        */

        if(nj1.EXTENSION.equals(".txt")){
            return pathname.getName().toLowerCase().endsWith(nj1.EXTENSION);
        }else if(nj1.EXTENSION.equals(".docx")){
            return pathname.getName().toLowerCase().endsWith(nj1.EXTENSION);
        }else if(nj1.EXTENSION.equals(".pptx")){
            return pathname.getName().toLowerCase().endsWith(nj1.EXTENSION);
        }else if(nj1.EXTENSION.equals(".xlsx")){
            return pathname.getName().toLowerCase().endsWith(nj1.EXTENSION);
        }else if(nj1.EXTENSION.equals(".doc")){
            return pathname.getName().toLowerCase().endsWith(nj1.EXTENSION);
        }else{
             Pattern p = Pattern.compile("\\.(txt|docx|xlsx|pptx|doc)$");
             String path = pathname.getName().toLowerCase();
             return p.matcher(path).find();
        }
```

จากนั้นผู้ใช้สามารถกดปุ่ม “เริ่มทำ Indexing” ก็จะมีผลลัพธ์การทำงานแสดงออกมา



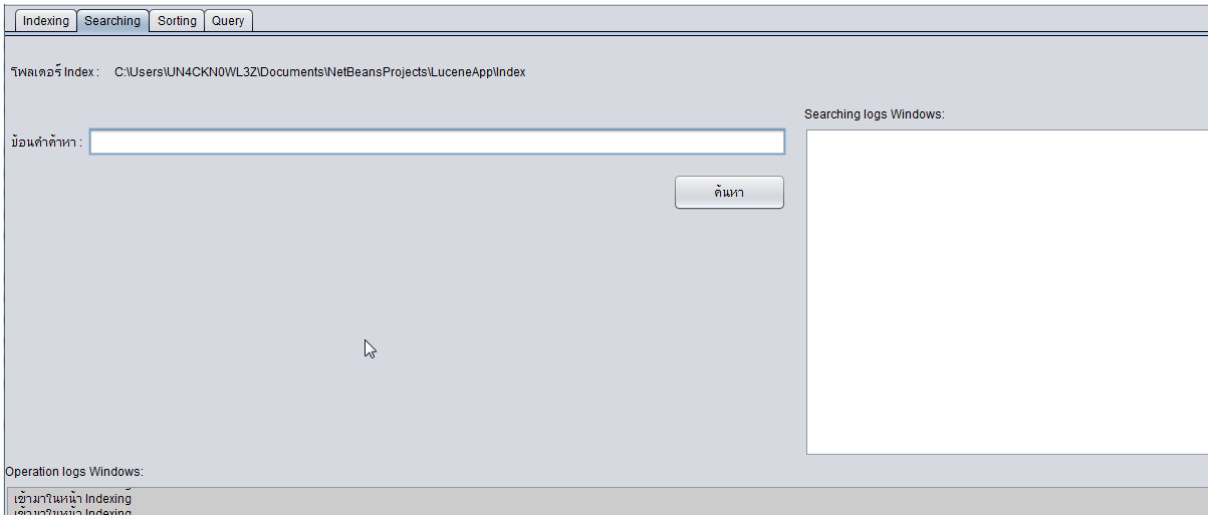

*โค้ดการทำงานในส่วนของ Button “เริ่มทำ Indexing”*

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
   // LuceneObj.StartIndex();
   if((dataPath=="" || dataPath=="") || (jLabel7.getText()=="-" || jLabel8.getText()=="-" ||jLabel7.getText()=="ยังไม่ได้เลือกโฟลเดอร์ Data" || jLabel8.getText(
       JOptionPane.showMessageDialog(null, "กรุณาเลือกโฟลเดอร์ Data และ โฟลเดอร์ Index!"
           ,"Error", JOptionPane.ERROR_MESSAGE);
       jTextArea1.setText(jTextArea1.getText() + "กรุณาเลือกโฟลเดอร์ Data และ โฟลเดอร์ Index!" +"\n");
   }else{
       //---------------------------------------------------
    LuceneTester LuceneObj;
    TextFileFilter ttfObj;
    try {
        LuceneObj = new LuceneTester();
        LuceneObj.dataDir = dataPath;
        LuceneObj.indexDir = indexPath;

    //    System.out.println(dataPath);
    //    System.out.println(indexPath);

        LuceneObj.createIndex();
        jTextArea2.setText("");
        jTextArea2.setText(LuceneObj.getIndexResult());
        jTextArea1.setText(jTextArea1.getText() + "ทำ Index เสร็จเรียบร้อย!" +"\n");
    } catch (IOException e) {
            //e.printStackTrace();
            JOptionPane.showMessageDialog(null, "มีบางอย่างผิดพลาด"
            ,"Error", JOptionPane.ERROR_MESSAGE);
            jTextArea1.setText(jTextArea1.getText() + "มีบางอย่างผิดพลาด" +"\n");
    }
   }
    //---------------------------------------------------
```

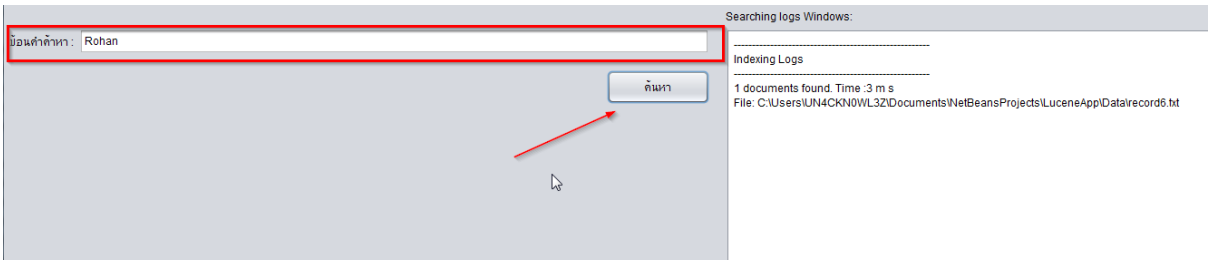
โปรแกรมจะมีการเรียกใช้คลาส LuceneTester และเรียกใช้ method createIndex() และมีการรับค่ากลับมาแสดงผลผ่าน method getIndexResult()

```
public void createIndex() throws IOException{
        indexer = new Indexer(indexDir);
        int numIndexed;
        long startTime = System.currentTimeMillis();
        numIndexed = indexer.createIndex(dataDir, new TextFileFilter());
        long endTime = System .currentTimeMillis();
        this.startTime = startTime;
        this.endTime = endTime;
        this.numIndexed = numIndexed;
        this.indexerLog = indexer.indexerLogs;
        indexer.close();
        //System.out.println(this.numIndexed+" File indexed, time taken: "+(this.startTime-this.endTime)+" m s");

}

public String getIndexResult(){

    return (this.indexerLog  +  "\n" + this.numIndexed+" File indexed, time taken: "+(this.endTime-this.startTime)+" m s");

}
```

## การทำ Searing




ก่อนที่จะทำ Searching ผู้ใช้ควรทำ Index มาก่อนเพราะจะมีการอ้างถึงโฟลเดอร์ Index การทำงานก็คือให้ผู้ป้อนคำที่ต้องการค้นหาลงใน Textbox




ก็จะมีการแสดงผลลัพธ์ออกทางหน้าจอ Searching Logs Windows ว่าเจอคำค้นอยู่ในไฟล์ไหน

โค้ดการทำงานใน Button “ค้นหา”

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(jTextField1.getText().equals("")){
         JOptionPane.showMessageDialog(null, "กรุณาป้อนคำค้น"
            ,"Alert!", JOptionPane.INFORMATION_MESSAGE);
       jTextArea1.setText(jTextArea1.getText() + "กรุณาป้อนคำค้น" +"\n");
    }else{
                    LuceneTester_Search tester;
            try {
                    tester = new LuceneTester_Search();
                    tester.indexDir = indexPath;
                    String term = jTextField1.getText();
                    tester.search(term);
                    jTextArea3.setText(tester.getSearchResult());
                    jTextArea1.setText(jTextArea1.getText() + "ทำ Search เสร็จเรียบร้อย!" +"\n");
            } catch (IO
                    e.p
            } catch (Pa
                    e.printStackTrace();
            }
    }
```

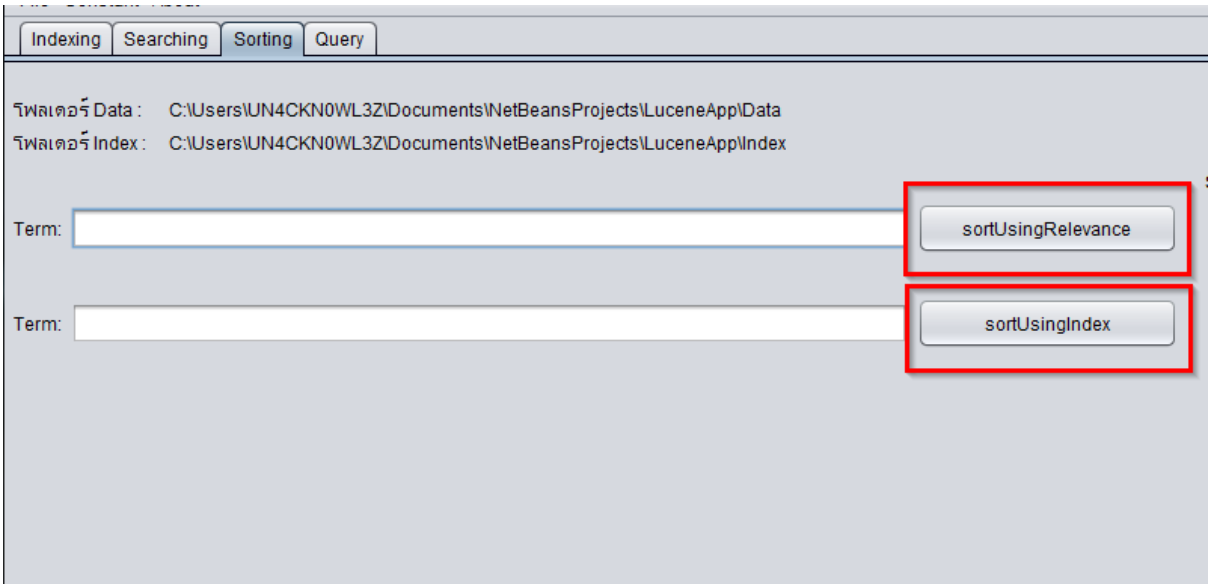
Print Stack Trace
---
(Alt-Enter shows hints)

มีการเรียกใช้คลาส LuceneTester_Search และทำการส่งค่าคำค้นที่เป็น String ไปยัง method search() และรับค่าผลลัพธ์การทำงานมาแสดงผลผ่าน method getSearchResult()

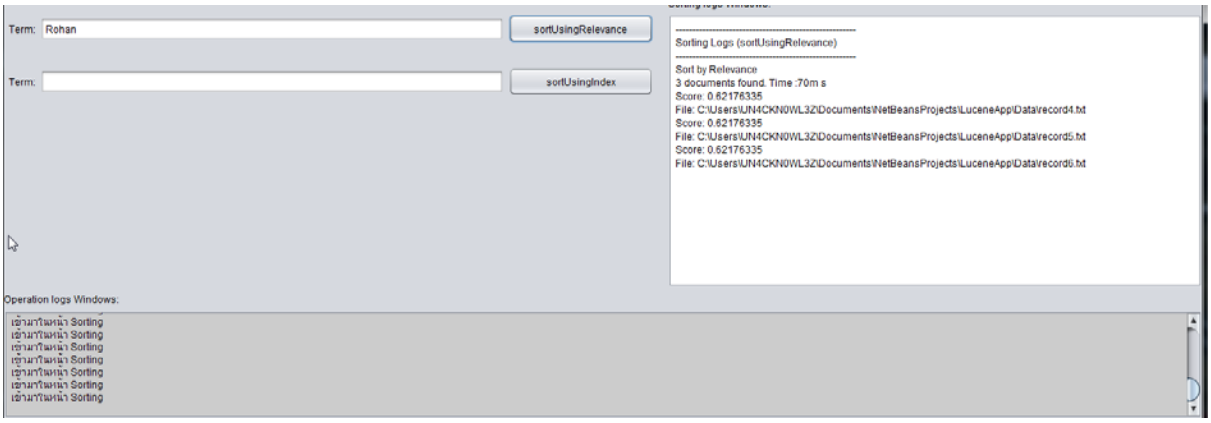
```
String indexDir ;
//String dataDir = "D:\\IR Material-20170911T002619Z-001\\IR Material\\Data1";

String searchResult;
Searcher searcher;

public void search(String searchQuery) throws IOException, ParseException{
        searcher = new Searcher(indexDir);
        long startTime = System .currentTimeMillis();
        TopDocs hits = searcher.search(searchQuery);
        long endTime = System .currentTimeMillis();
        //System.out.println(hits.totalHits +" documents found. Time :" + (endTime - startTime) +" m s");
        this.searchResult = hits.totalHits +" documents found. Time :" + (endTime - startTime) +" m s"+"\n";
        for(ScoreDoc scoreDoc : hits.scoreDocs) {
                Document doc = searcher.getDocument(scoreDoc);
                //System.out.println("File: "+ doc.get(LuceneConstants.FILE_PATH));
                this.searchResult += "File: "+ doc.get(LuceneConstants.FILE_PATH)+"\n";
        }
        searcher.close();
}

public String getSearchResult(){

    return "-------------------------------------------------------\n"+"Indexing Logs\n"+"--------------------------

}
```

**การทำ Sorting**

การทำ Sorting จะประกอบไปด้วย 2 method คือ sortUsingRelevance() และ sortUsingIndex() มีการแยกส่วนสำหรับการใช้งาน 2 method นี้



ผลลัพธ์การทำงานของ method sortUsingRelevance()




*โค้ดการทำงาน*

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
            if(jTextField2.getText().equals("")){
          JOptionPane.showMessageDialog(null, "กรุณาป้อนคำค้น"
            ,"Alert!", JOptionPane.INFORMATION_MESSAGE);
       jTextArea1.setText(jTextArea1.getText() + "กรุณาป้อนคำค้น" +"\n");
    }else{

            LuceneTester_Sort testerS;
            try {
                    testerS = new LuceneTester_Sort();
                    testerS.dataDir = dataPath;
                    testerS.indexDir = indexPath;
                    testerS.sortUsingRelevance(jTextField2.getText());
                    jTextArea4.setText(testerS.getResultRelevance());
                    jTextArea1.setText(jTextArea1.getText() + "ทำ Sorting (sortUsingRelevance) เสร็จเรียบร้อย!" +"\n");
            } catch (IOException e) {
                    e.printStackTrace();
            } catch (ParseException e) {
                    e.printStackTrace();
            }
    }
```

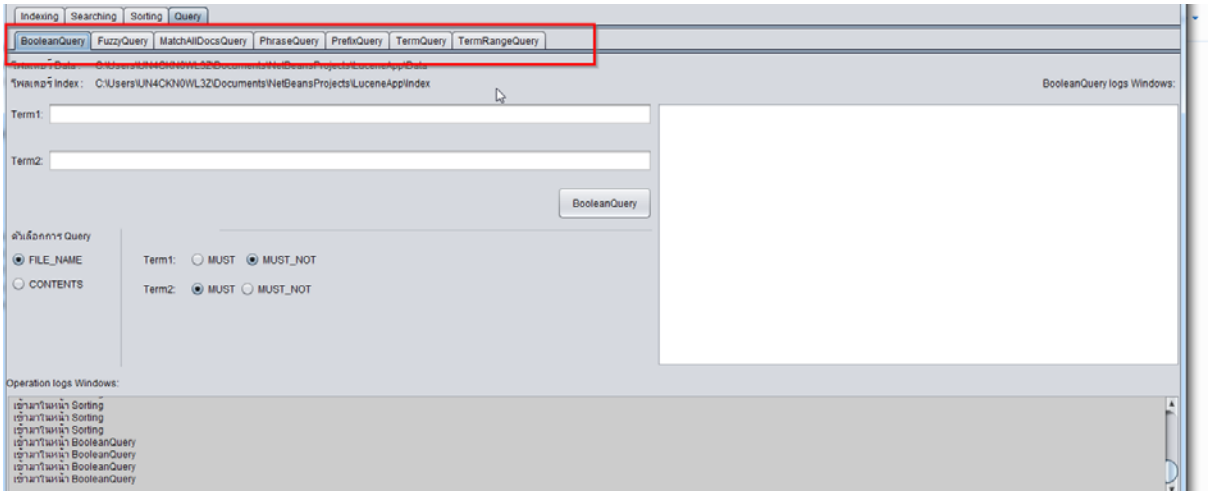
มีการเรียกไปยังคลาส LuceneTester_Sort และส่งค่าไปยัง method sortUsingRelevance() เมื่อโปรแกรมทำงานเสร็จก็จะรับค่ามาแสดงผลผ่าน method getResultRelevance()

```
*/
    public void sortUsingRelevance(String searchQuery)throws IOException, ParseException{
            //System.out.println("Sort by Relevance");
            this.sortingResultRelevance +="Sort by Relevance\n";
            searcher = new Searcher(indexDir);
            long startTime = System.currentTimeMillis();
            //create a term to search file nam e
            Term term = new Term (LuceneConstants.CONTENTS, searchQuery);
            //create the term query object
            Query query = new FuzzyQuery(term );
            searcher.setDefaultFieldSortScoring(true, false);
            //do the search
            TopDocs hits = searcher.search(query,Sort.RELEVANCE);
            long endTime = System.currentTimeMillis();
            //System.out.println(hits.totalHits +" documents found. Time :" + (endTime - startTime) + "m s");
            this.sortingResultRelevance += hits.totalHits +" documents found. Time :" + (endTime - startTime) + "m s\n";
            for(ScoreDoc scoreDoc : hits.scoreDocs) {
                    Document doc = searcher.getDocument(scoreDoc);
                    //System.out.print("Score: "+ scoreDoc.score + " ");

                    this.sortingResultRelevance += "Score: "+ scoreDoc.score + " \n";

                    //System.out.println("File: "+ doc.get(LuceneConstants.FILE_PATH));

                    this.sortingResultRelevance += "File: "+ doc.get(LuceneConstants.FILE_PATH)+"\n";
            }
            searcher.close();
    }
```

การทำงานของ method sortUsingIndex() ก็ทำงานในลักษณะเดียวกัน

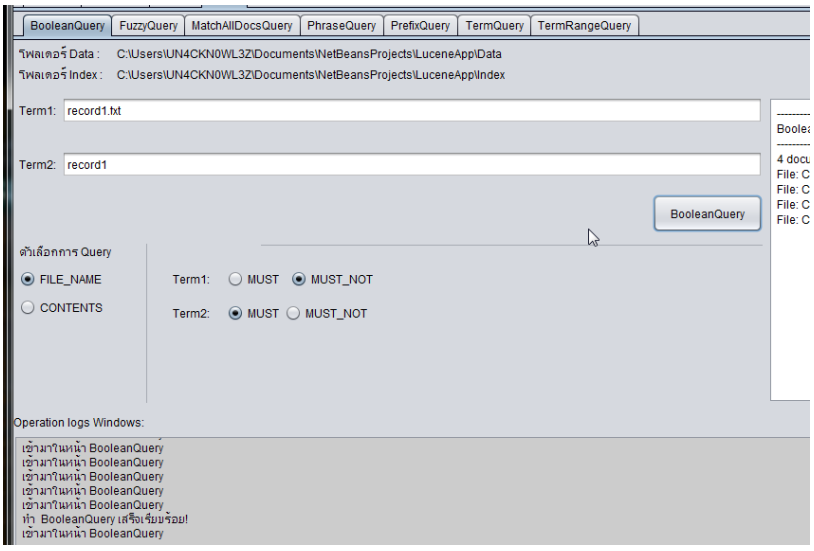
**การ Query**

ในส่วนของการทำ Quey ตัวโปรแกรมได้แบ่งเป็น method ย่อยๆดังนี้ BooleanQuery, FuzzyQuery, MatchAllDocsQuery, PhraseQuery, PrefixQuery, TermQuery, TermRangeQuery

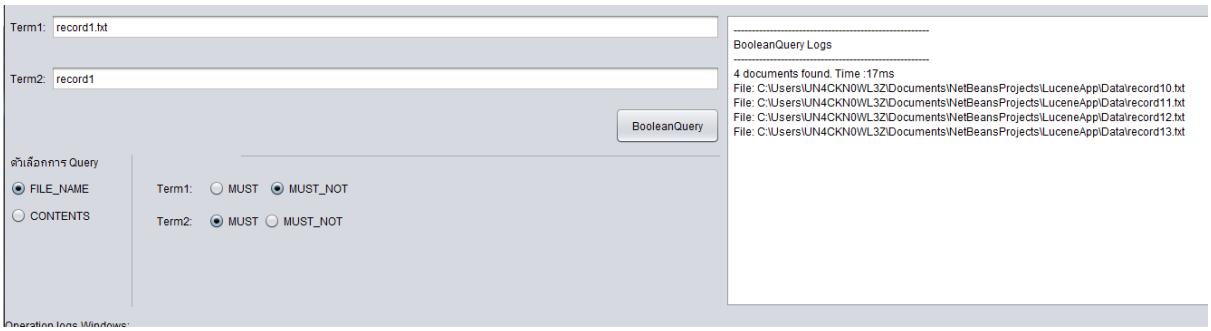



**BooleanQuery**

ในส่วนของ BooleanQuery จะมีตัวเลือกการทำงานด้วย จะมีอยู่ 2 ส่วนคือ จะเลือกว่างานในใน FILE_NAME หรือใน CONTENTS และ เลือกการ Occur ของ Term ว่าจะ MUST หรือ MUST_NOT



เมื่อโปรแกรมทำงานก็จะมีผลลัพธ์ออกมาตามที่เซ็ตไว้




*โค้ดการทำงานใน Button “BooleanQuery”*

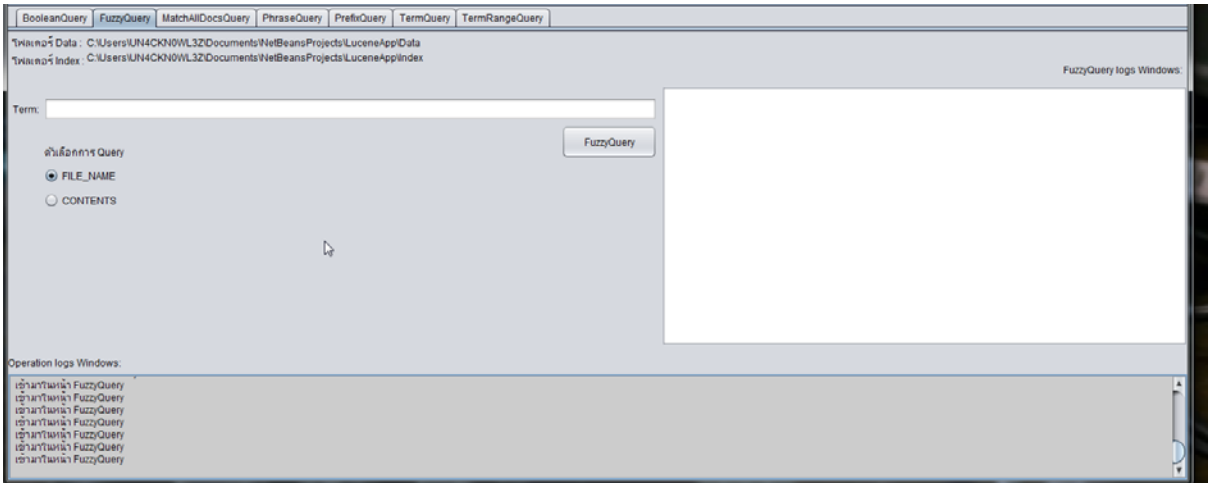
```
                    if(jTextField4.getText().equals("") || jTextField5.getText().equals("")){
                JOptionPane.showMessageDialog(null, "กรุณาป้อนคำค้น ทั้ง 2 คำ"
                    ,"Alert!", JOptionPane.INFORMATION_MESSAGE);
               jTextArea1.setText(jTextArea1.getText() + "กรุณาป้อนคำค้น ทั้ง 2 คำ" +"\n");
            }else{
                      LuceneTester_BooleanQuery testerBL;
                    try {
                            testerBL = new LuceneTester_BooleanQuery();
                            testerBL.dataDir = dataPath;
                            testerBL.indexDir = indexPath;
                            if(jRadioButton11.isSelected()){
                                testerBL.queryOptions = true;
                            }else{
                            testerBL.queryOptions = false;
                               }

                            if(jRadioButton13.isSelected()){
                                testerBL.mustOrNot1 = true;
                            }else{
                            testerBL.mustOrNot1 = false;
                               }

                             if(jRadioButton15.isSelected()){
                                testerBL.mustOrNot2 = true;
                            }else{
                            testerBL.mustOrNot2 = false;
                               }

                           testerBL.searchUsingBooleanQuery(jTextField4.getText(), jTextField5.getText());
                            jTextArea5.setText(testerBL.getResultBooleanQuery());
                            jTextArea1.setText(jTextArea1.getText() + "ทำ  BooleanQuery เสร็จเรียบร้อย!" +"\n");
                    } catch (IOException e) {
                            e.printStackTrace();
                    } catch (ParseException e) {
```

มีการเรียกไปยังคลาส LuceneTester_BooleanQuery และส่งค่าตัวเลือกต่างๆและ return ผลลัพธ์กลับมาแสดงผล

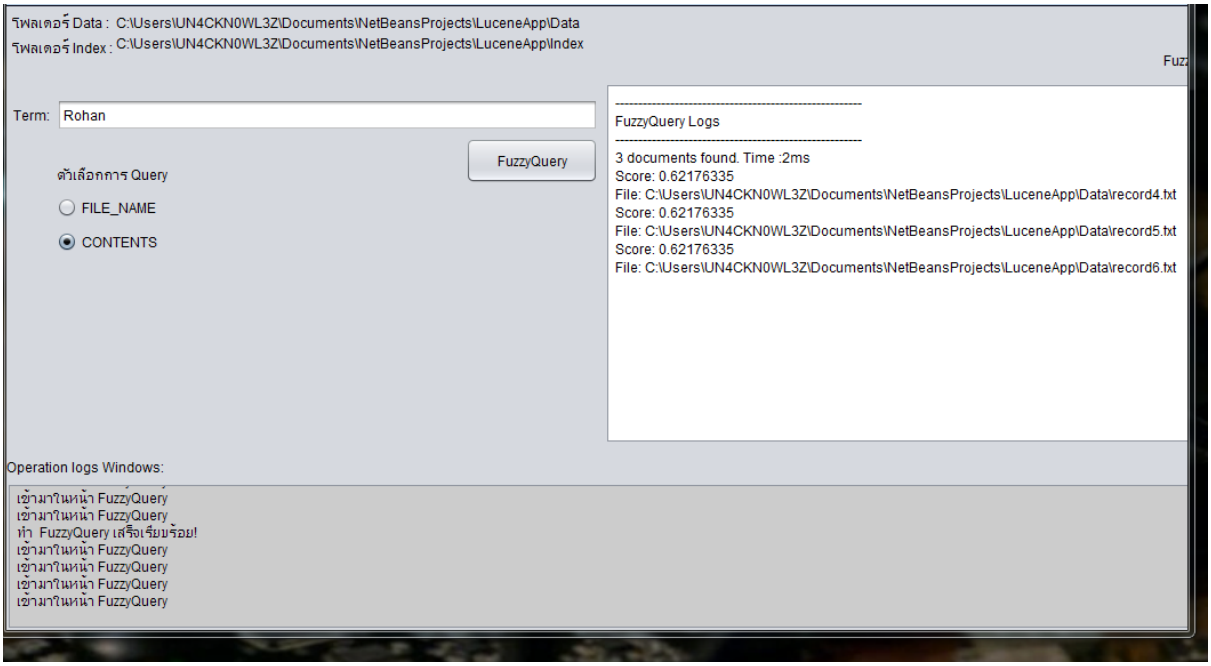
```
String indexDir ;
String dataDir ;
Searcher searcher;
String resultBooleanQuery="";
boolean queryOptions;
boolean mustOrNot1;
boolean mustOrNot2;
Term term1;
Term term2;
public void searchUsingBooleanQuery(String searchQuery1,
   String searchQuery2)throws IOException, ParseException{
   searcher = new Searcher(indexDir);
   long startTime = System.currentTimeMillis();
   //create a term to search file name

   if(queryOptions){
       Term term1 = new Term(LuceneConstants.FILE_NAME, searchQuery1);
       Term term2 = new Term(LuceneConstants.FILE_NAME, searchQuery2);
       this.term1 = term1;
       this.term2 = term2;
   }else{
       Term term1 = new Term(LuceneConstants.CONTENTS, searchQuery1);
       Term term2 = new Term(LuceneConstants.CONTENTS, searchQuery2);
       this.term1 = term1;
       this.term2 = term2;
   }
```

```
//create the term query object
Query query2 = new PrefixQuery(term2);

BooleanQuery query = new BooleanQuery();

if(mustOrNot1){
    query.add(query1,BooleanClause.Occur.MUST);
}else{
    query.add(query1,BooleanClause.Occur.MUST_NOT);
}

if(mustOrNot2){
    query.add(query2,BooleanClause.Occur.MUST);
}else{
    query.add(query2,BooleanClause.Occur.MUST_NOT);
}


//do the search
TopDocs hits = searcher.search(query);
long endTime = System.currentTimeMillis();

// System.out.println(hits.totalHits +
//       " documents found. Time :" + (endTime - startTime) + "ms");

this.resultBooleanQuery += hits.totalHits +" documents found. Time :" + (endTime - startTime) + "ms\n";
for(ScoreDoc scoreDoc : hits.scoreDocs) {
   Document doc = searcher.getDocument(scoreDoc);
   // System.out.println("File: "+ doc.get(LuceneConstants.FILE_PATH));
   this.resultBooleanQuery += "File: "+ doc.get(LuceneConstants.FILE_PATH)+"\n";
}
searcher.close();
```

## FuzzyQuery




มีลักษะการทำงานเหมือนกับ BooleanQuery แต่จะตัดตัวเลือกการ Occur ออไป



*โค้ดการทำงานใน Button FuzzyQuery*

```
private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    if(jTextField6.getText().equals("")){
         JOptionPane.showMessageDialog(null, "กรุณาป้อนคำค้น"
            ,"Alert!", JOptionPane.INFORMATION_MESSAGE);
       jTextArea1.setText(jTextArea1.getText() + "กรุณาป้อนคำค้น" +"\n");
    }else{

           LuceneTester_FuzzyQuery testerFQ;
           try {
            testerFQ = new LuceneTester_FuzzyQuery();
            testerFQ.dataDir = dataPath;
            testerFQ.indexDir = indexPath;
          //      System.out.println(jRadioButton1.isSelected());
            if(jRadioButton1.isSelected()){
                testerFQ.queryOptions = true;
            }else{
                testerFQ.queryOptions = false;
            }
            testerFQ.searchUsingFuzzyQuery(jTextField6.getText());
            jTextArea6.setText(testerFQ.getResultFuzzyQuery());
            jTextArea1.setText(jTextArea1.getText() + "ทำ  FuzzyQuery เสร็จเรียบร้อย!" +"\n");
            } catch (IOException e) {
                e.printStackTrace();
            } catch (ParseException e) {
                e.printStackTrace();
            }
```

*โค้ดการทำงานในคลาส LuceneTester_FuzzyQuery*

```
*/
public void searchUsingFuzzyQuery(String searchQuery)
   throws IOException, ParseException{
   searcher = new Searcher(indexDir);
   long startTime = System.currentTimeMillis();
   //create a term to search file name
  // System.out.println(queryOptions);
   if(queryOptions){
       Term term = new Term(LuceneConstants.FILE_NAME, searchQuery);
       this.term = term;
   }else{
       Term term = new Term(LuceneConstants.CONTENTS, searchQuery);
        this.term = term;
   }
 //  Term term = new Term(LuceneConstants.FILE_NAME, searchQuery);
   //create the term query object

   Query query = new FuzzyQuery(term);
   //do the search
   TopDocs hits = searcher.search(query);
   long endTime = System.currentTimeMillis();

  // System.out.println(hits.totalHits +" documents found. Time :" + (endTime - startTime) + "ms");
   this.resultFuzzyQuery += hits.totalHits + " documents found. Time :" + (endTime - startTime) + "ms\n";

   for(ScoreDoc scoreDoc : hits.scoreDocs) {
      Document doc = searcher.getDocument(scoreDoc);
     // System.out.print("Score: "+ scoreDoc.score + " ");
      this.resultFuzzyQuery += "Score: "+ scoreDoc.score + " \n";
     // System.out.println("File: "+ doc.get(LuceneConstants.FILE_PATH));
      this.resultFuzzyQuery += "File: "+ doc.get(LuceneConstants.FILE_PATH)+"\n";
```

ในการทำ Query รูปแบบอื่นๆก็จะทำงานในลักษณะเดียวกัน