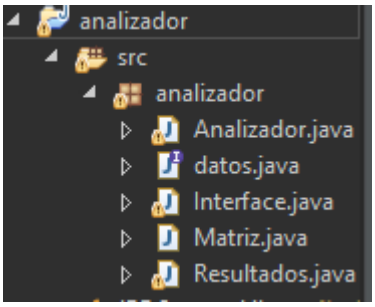
# Manual técnico

## Configuración del entorno de desarrollo JAVA

Para el desarrollo de la aplicación se utilizó el lenguaje de tercera generación JAVA, por lo mismo, es necesario tener en el equipo la versión más actualizada de JDK.

**Nota:** Es necesario tener instalado el JDK, en caso de no contar con este puede descargarse desde la página http://www.oracle.com.

**Nota:** El proyecto fue desarrollado en Eclipse, pero se puede utilizar cualquier IDE de Java.




**Estructura de Archivos**

- **Analizador:** Es la clase principal del proyecto.
- **Datos:** Es una interface la cual se usará para almacenar las estadísticas para luego ser procesadas por Resultados.java
- **Interface:** Clase que hereda a Datos
- **Matriz:** En ella se encuentra los signos permitidos en nuestro lenguajes
- **Resultados:** La clase la cual mostrara los resultados del análisis

```java
private void Palabras(){
     palabras= new ArrayList<String>();
     String texto=editorPane.getText().toString();
     String palabra="";

    int contador=0;
    int findeLinea=texto.length()-1;
    int iniciop=0;
    int finalp=0;
     cant_letras=0;
    boolean estado=false;
    for(int i=0;i<texto.length();i++){
        if(Character.isLetterOrDigit(texto.charAt(i))&& i!=findeLinea){
            estado=true;
            finalp++;
            palabra=palabra+String.valueOf(texto.charAt(i));
        }else if(!(Character.isLetterOrDigit(texto.charAt(i))) && estado){
            estado=false;
            cant_letras=cant_letras+(finalp-iniciop);
                iniciop=finalp;
                    contador++;
                    palabras.add(palabra);
                palabra="";
        }else if(Character.isLetterOrDigit(texto.charAt(i)) && i==findeLinea){
            cant_letras=cant_letras+(finalp+1-iniciop);
            contador++;
            palabra=palabra+String.valueOf(texto.charAt(i));
            palabras.add(palabra);
            palabra="";
        }
    }
    cantidad.add(contador);
```

Cada tipo de análisis tendrá su propia función y/o método, en este, se cuenta la cantidad de letras y palabras que hay en el texto ingresando, recorriendo el texto con un for, si se encuentra un espacio, se guardar la palabra en un arraylist para su posterior análisis.

```java
public void Monosilaba(){
    int cantidad_mono=0;
    int contador=0;
    for(int j=0;j<palabras.size();j++){
        String palabra=palabras.get(j);
        if(!esHiato(palabra)){
            contador=0;
            for(int i=0;i<palabra.length();) {
                if(Character.isLetter(palabra.charAt(i))) {
                    try{
                        if(  (!(esVocal(palabra.charAt(i))) && esVocal(palabra.charAt(i+1))) ||  (esVocal(palabra.charAt(i)) && !(esVocal(palabra.ch
                            contador++;                     }
                    }catch(StringIndexOutOfBoundsException error) {}
                }
                i+=2;
                }
        }else{
            contador=2;
        }
        if(contador>1) {
        }else {
            cantidad_mono++;
        }

    }
    cantidad.add(cantidad_mono);

}

public  boolean esVocal(char c){
    String vocal="aeiouAEIOU";
    if(vocal.indexOf(c)!=-1) return true;
    else return false;
}
```

Para verificar si una palabra es monosílaba, se ingresa al proceso, donde primero se comprueba si es hiato, si esta no es hiato, sigue en el proceso, donde se verifica que el carácter de la palabra sea una letra, luego si esta la combinación vocal+ consonante o consonante + vocal , más de una vez en la palabra se concluye que no es una monosílaba.

```java
    mntmAbrir.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            JFileChooser abrir=new JFileChooser();
            BufferedReader br;
            try{
                if(abrir.showOpenDialog(null)==abrir.APPROVE_OPTION){
                    ruta=abrir.getSelectedFile().getAbsolutePath();
                    File fileDir = new File(ruta);
                    br = new BufferedReader(new InputStreamReader(new FileInputStream(fileDir), "UTF-8"));
                    editorPane.read(br, null);

                }
            }catch(Exception ex){
                ex.printStackTrace();
            }
        }
    });
    mnMenuArchivo.add(mntmAbrir);
//----------------------------------------------------------------------Guardar Archivo----------------------------------------------------------------
    JMenuItem mntmGuardar = new JMenuItem("Guardar");
    mntmGuardar.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
             try(FileOutputStream fos=new FileOutputStream(ruta)){
                    String texto=editorPane.getText().toString();
                    byte codigos[]=texto.getBytes();
                    fos.write(codigos);
                }catch(IOException ee){}

        }
    });
    mnMenuArchivo.add(mntmGuardar);
//----------------------------------------------------------------------Guardar Como-------------------------------------------------------------------
```

En la función de abrir y guardar, se una la función FileInputStream para guardar el archivo en modo “UTF-8” para así poder guardar todo tipo de caracteres sin excepción.

```java
private void Guardar(){
    Interface inter= new Interface();
    inter.setCantidad(cantidad);
    inter.setCant_errores(cant_errores);
    inter.setCant_letras(cant_letras);
    inter.setCorrecion(correcion);

    Resultados resultados= new Resultados();
    resultados.setVisible(true);
}
```

Para guardar los datos para posteriormente utilizarlos en Resultados.java se instancia la clase interface y se procede a guardar los datos.