¥ø

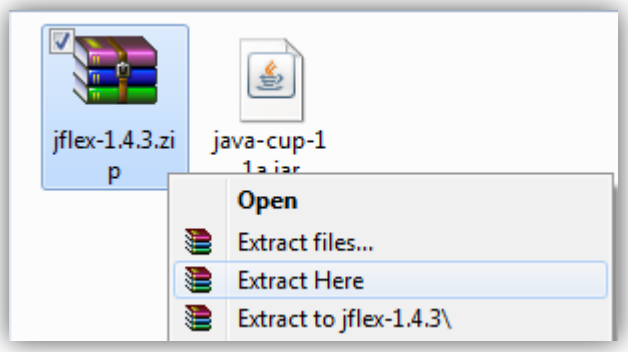
Tutorial: JFlex, Cup y NetBeans

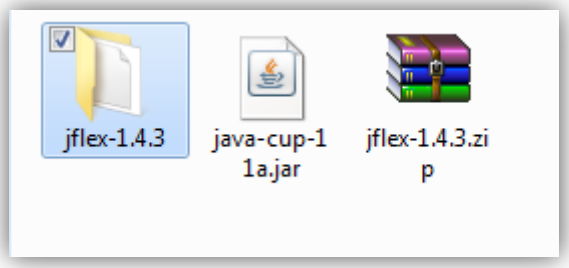
El siguiente tutorial está dirigido para los que están trabajando por primera vez con las herramientas JFlex y Cup y desean utilizarlas con NetBeans.

Prerrequisitos:

1) Tener las librerías de java. Pueden descargar el archivo jdk-6u18-windows-i586.exe en la página http://java.sun.com/javase/downloads/widget/jdk6.jsp.
2) Tener instalado NetBeans. Pueden descargarlo en la página http://netbeans.org/downloads/index.html.
3) Tener JFlex. Pueden descargar el archivo jflex-1.4.3.zip en la página http://jflex.de/download.html.
4) Tener Cup. Pueden descargar el archivo java-cup-11a.jar en la página http://www2.cs.tum.edu/projects/cup/.

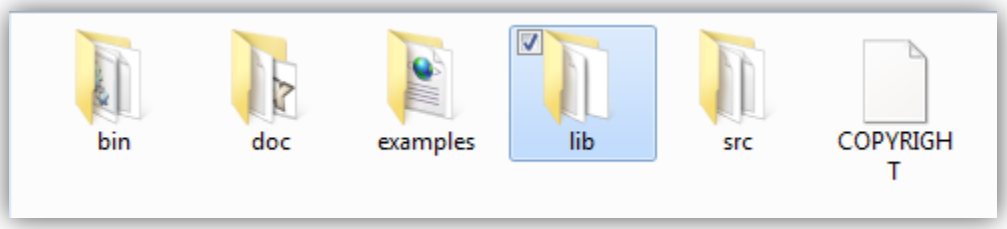
Ya teniendo estos elementos en nuestra computadora, procedemos a descomprimir el archivo de jflex-1.4.3.zip

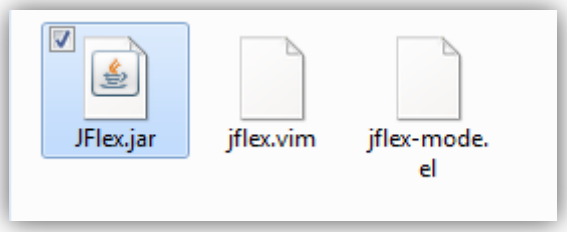




¥ø

Entramos y buscamos la carpeta lib dentro de la cual estará un archivo llamado JFlex.jar

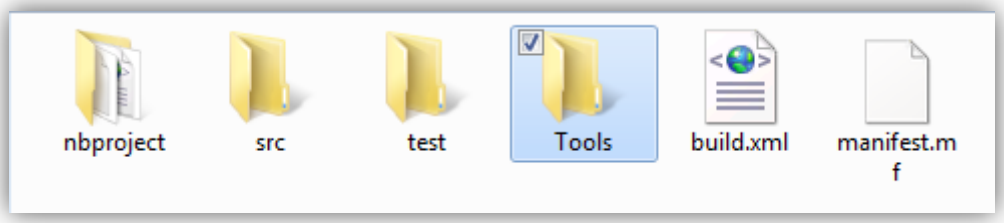




Para la integración con el NetBeans solo ocupamos el archivo de JFlex.jar y el java-cup-11a.jar al cual le podemos cambiar el nombre a Cup.jar sin problema alguno.

Para la integración tenemos varias opciones:

1) Si solo planean usarlo en un proyecto, es recomendable copiar los archivos en una carpeta x dentro del proyecto.

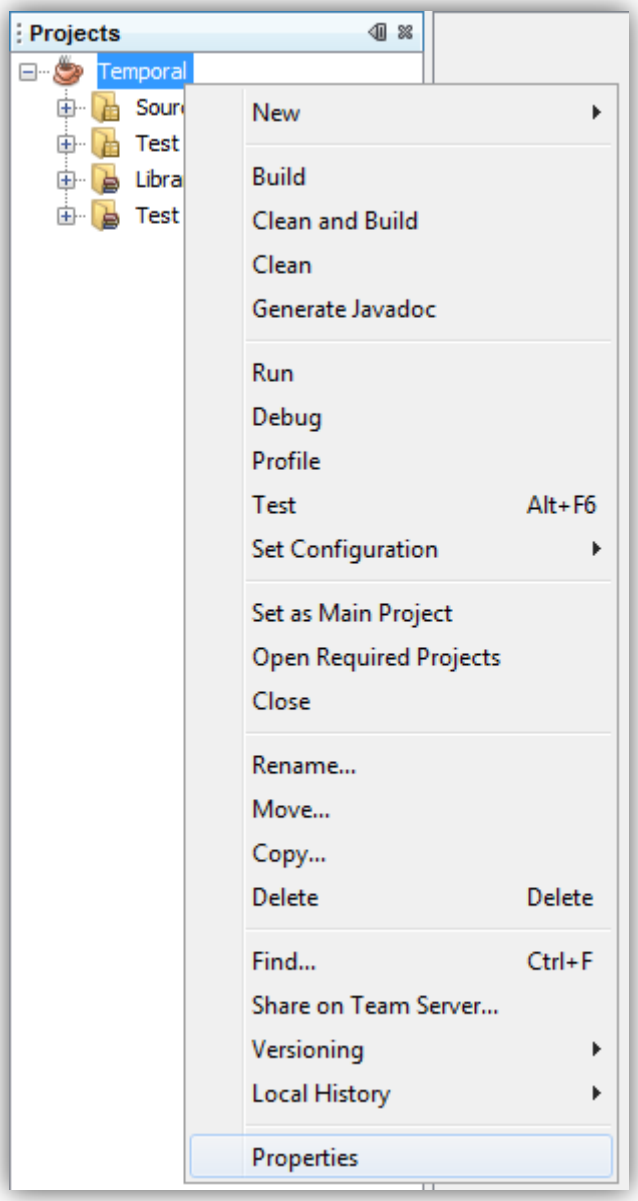


Recordando que es un proyecto de NetBeans.

2) Pero si planean usarlo para varios proyectos es mejor crear la carpeta en el disco local. Para este tutorial solo lo haremos para un proyecto.

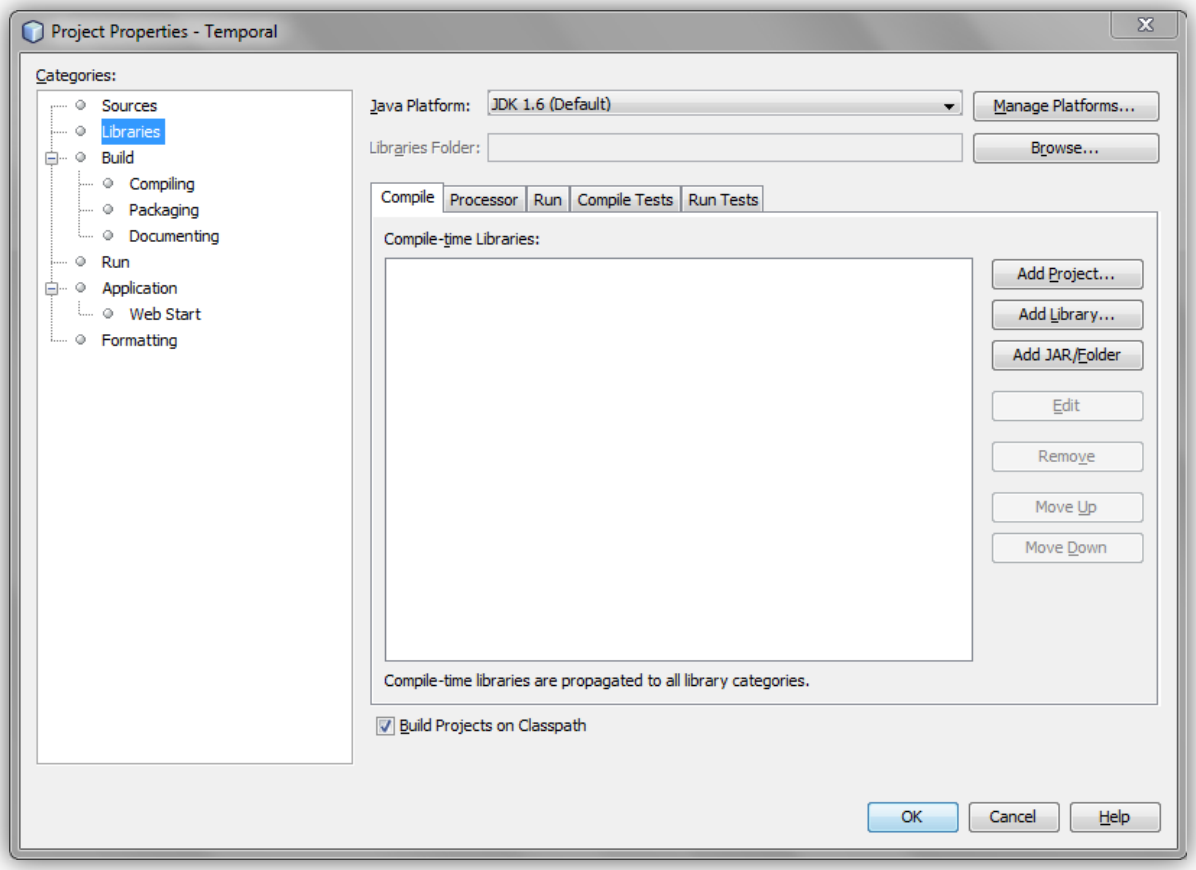
¥ø

Ya habiendo creado el proyecto y teniendo la carpeta Tools adentro con los correspondientes archivos .jar nos pasamos a NetBeans y en el proyecto le damos click secundario y seleccionamos propiedades.



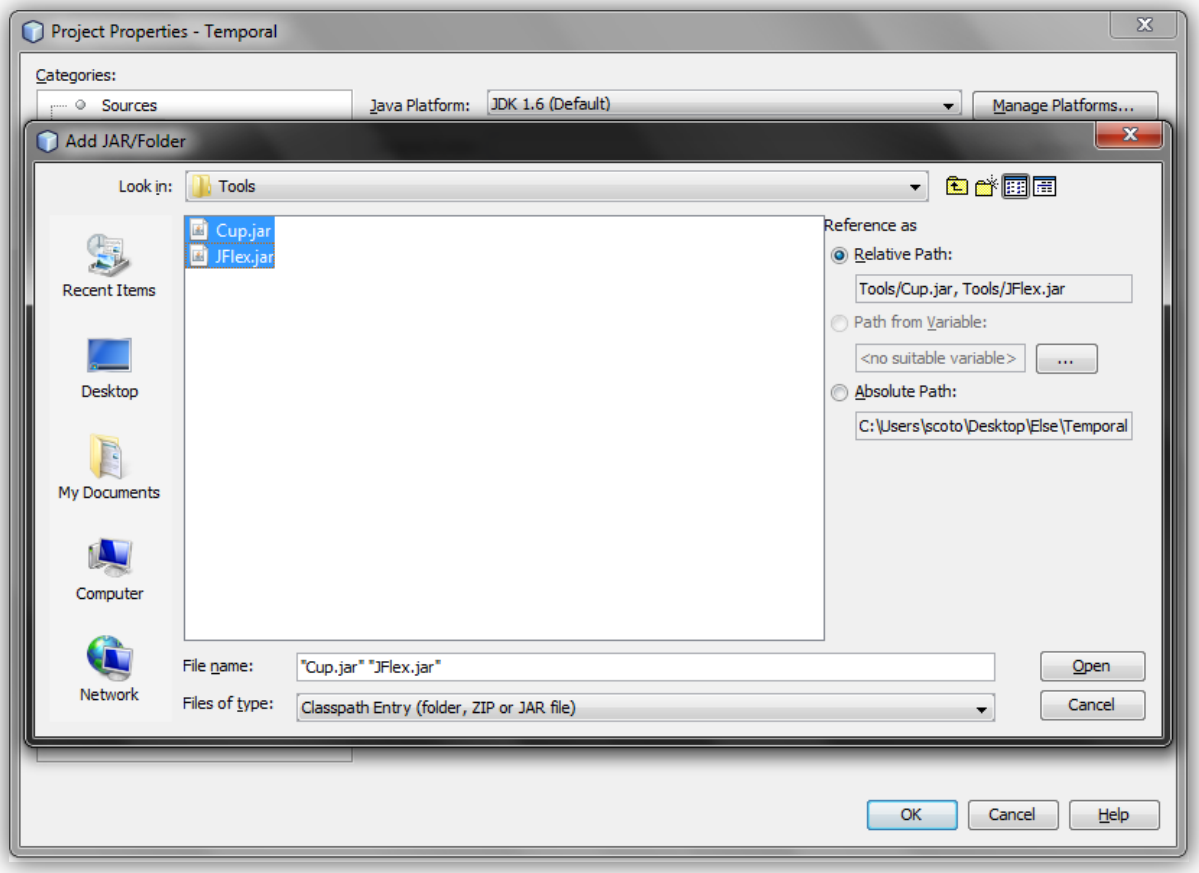
¥ø

Luego seleccionamos donde dice librerías

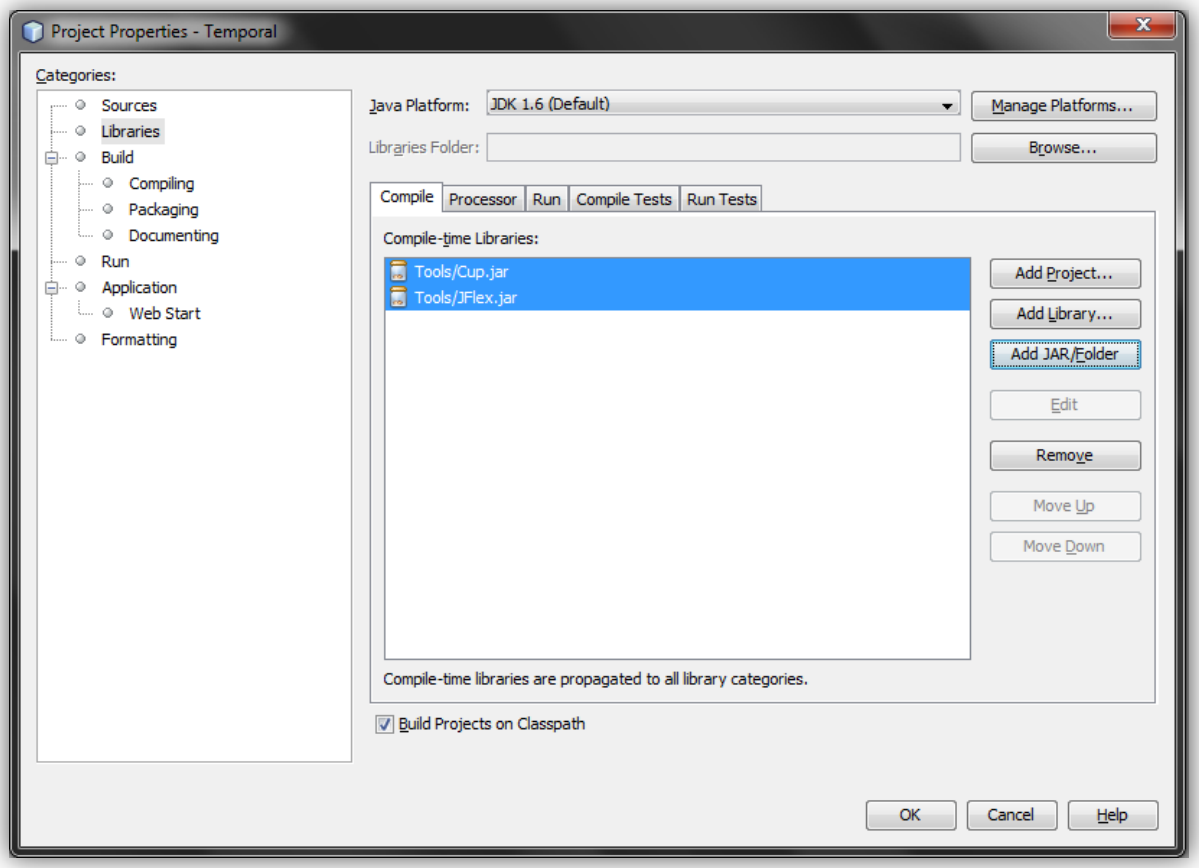


¥ø

Y cliqueamos en la opción de agregar un jar/folder. Buscamos la ubicación de la carpeta Tools y seleccionamos los dos archivos jar.



¥ø



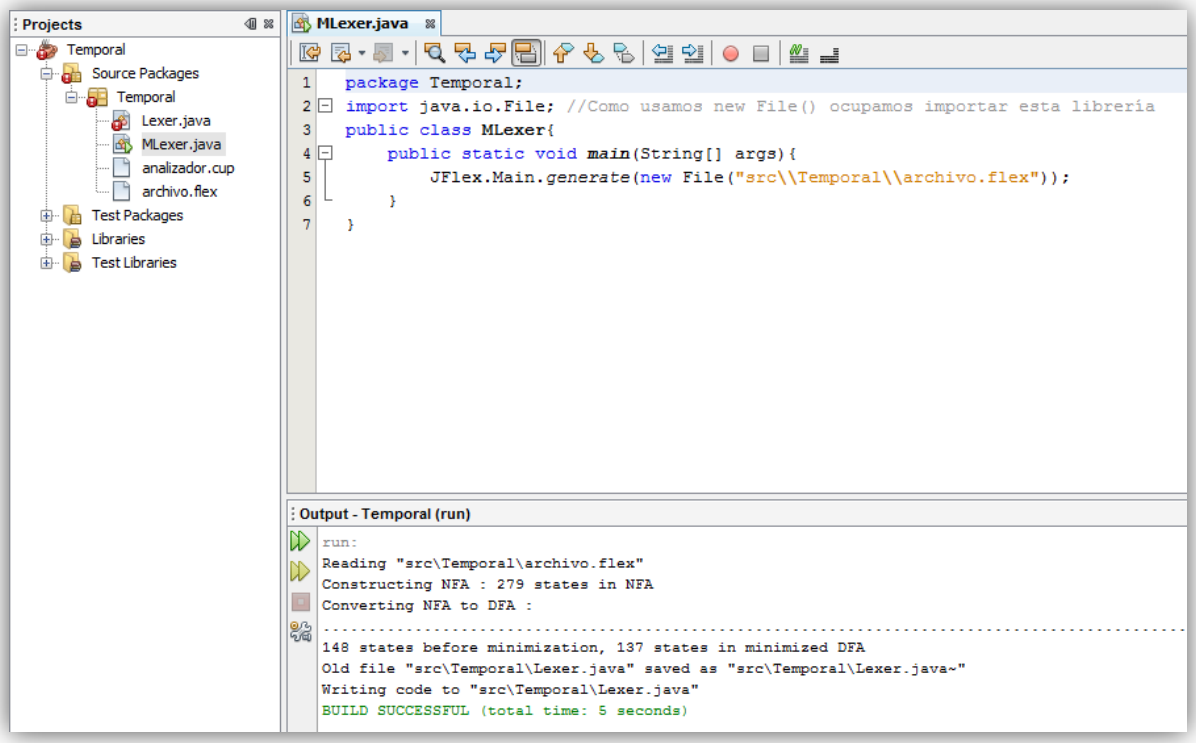
Con esto ya tenemos agregados las herramientas al NetBeans. Para poder usarlas solo ocupamos el siguiente código:

Codigo: MLexer

```
package Temporal;

import java.io.File; //Como usamos new File() ocupamos importar esta librería

public class MLexer{

   public static void main(String[] args){

      JFlex.Main.generate(new File("src\\Temporal\\archivo.flex"));

   }

}

//Donde archvio.flex es su archivo .flex
```

Lo corren y les generara un archivo llamado Lexer.java

¥ø



Cuando utilizamos el cup podemos decidir sobre varios aspectos:

-package name specify package generated classes go in [default none]

-destdir name specify the destination directory, to store the generated files in

-parser name specify parser class name [default "parser"]

-typearg args specify type arguments for parser class

-symbols name specify name for symbol constant class [default "sym"]

-interface put symbols in an interface, rather than a class

-nonterms put non terminals in symbol constant class

-expect # number of conflicts expected/allowed [default 0]

-compact_red compact tables by defaulting to most frequent reduce

-nowarn don't warn about useless productions, etc.

-nosummary don't print the usual summary of parse states, etc.

-nopositions don't propagate the left and right token position values

-noscanner don't refer to java_cup.runtime.Scanner

-progress print messages to indicate progress of the system

-time print time usage summary

¥ø

-dump_grammar produce a human readable dump of the symbols and grammar

-dump_states produce a dump of parse state machine

-dump_tables produce a dump of the parse tables

-dump produce a dump of all of the above

-version print the version information for CUP and exit

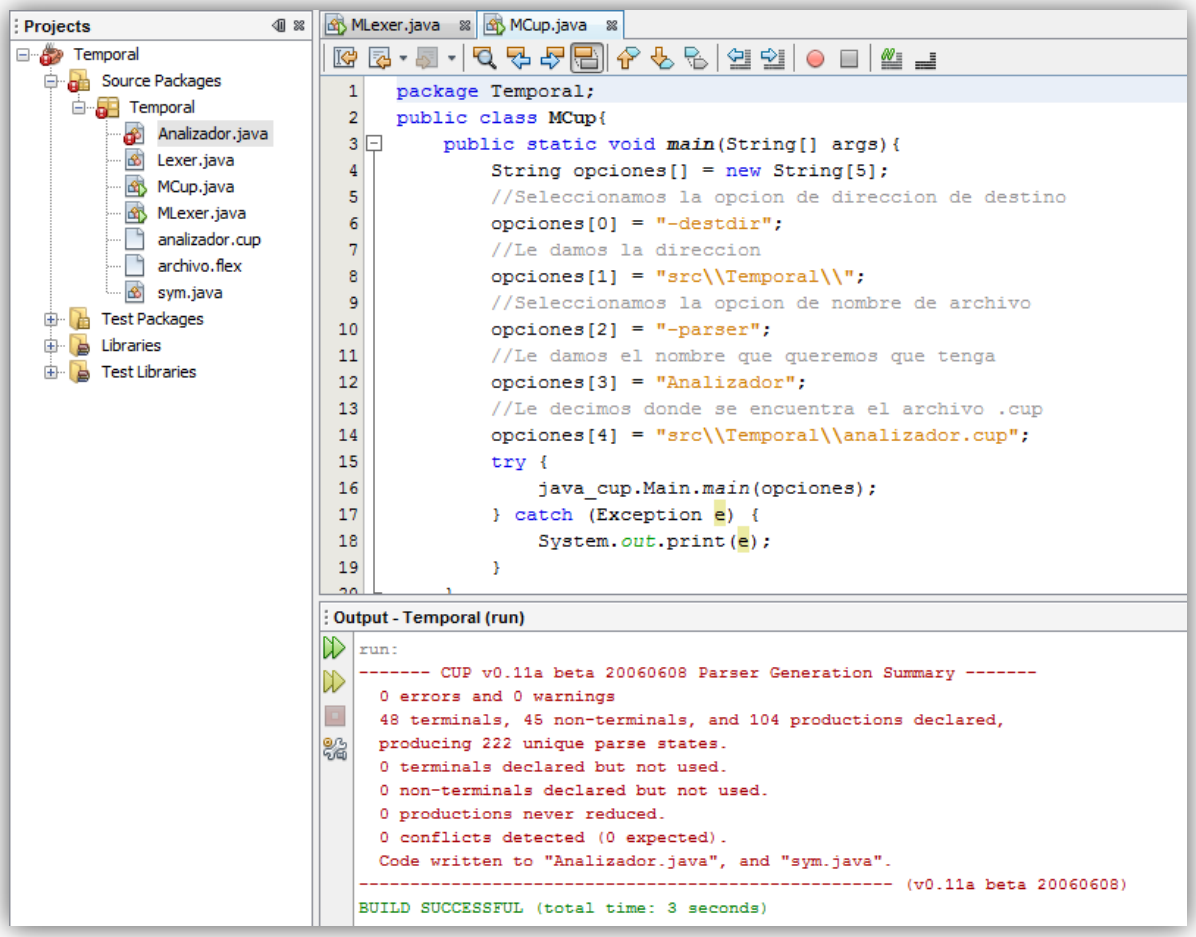
Para este ejemplo solo utilizaremos las opciones de destino y de nombre.

Codigo: MCup

```
package Temporal;

public class MCup{
   public static void main(String[] args){
      String opciones[] = new String[5];
      //Seleccionamos la opción de dirección de destino
      opciones[0] = "-destdir";
      //Le damos la dirección
      opciones[1] = "src\\Temporal\\";
      //Seleccionamos la opción de nombre de archivo
      opciones[2] = "-parser";
      //Le damos el nombre que queremos que tenga
      opciones[3] = "Analizador";
      //Le decimos donde se encuentra el archivo .cup
      opciones[4] = "src\\Temporal\\analizador.cup";
      try {
         java_cup.Main.main(opciones);
      } catch (Exception e) {
         System.out.print(e);
      }
```

¥ø

```
    }

}
```



Lo cual nos generara 2 archivos .java, uno es el Analizador.java o el nombre que hayan elegido y el otro es sym.java.