# Práctica 2: un traductor descendente implementado a máquina

### Procesadores de Lenguaje, curso 2012-2013

Esta práctica consiste en implementar un traductor similar al de la práctica 1. En este caso, sin embargo, la implementación del analizador léxico y del analizador sintáctico no se realiza *a mano*, sino *a máquina* mediante el programa ANTLR, que es una herramienta de código abierto que simplifica la construcción de traductores entre lenguajes informáticos porque genera automáticamente el analizador léxico y el sintáctico a partir de sus especificaciones formales.

## Consideraciones generales

1. Ten en cuenta que, a partir de este curso, **las prácticas serán individuales**.
2. En la sección de materiales del Campus Virtual podrás encontrar unos cuantos ficheros de prueba junto a un programa que determina la corrección de la salida de tu práctica con ellos.
3. Has de entregar un único fichero, llamado plp2.tgz, que cumpla con las especificaciones de este enunciado.
4. Las fechas de entrega de esta práctica son:

    - 1.ª entrega: viernes 18 de enero de 2013, hasta las 23:59
    - 2.ª entrega: viernes 1 de febrero de 2013, hasta las 23:59  (retrasada una semana por coincidir con los exámenes)

### Características del código entregado

El archivo tgz que entregues ha de contener exclusivamente el código fuente (con extensión .java) de todas las clases que hayas implementado para tu traductor, además del fichero con la especificación para ANTLR, que ha de llamarse forzosamente plp2.g. Los ficheros del archivo tgz no pueden estar contenidos dentro de ningún directorio. Además, tus clases no pueden pertenecer a ningún paquete ni importar ninguna librería de Java que no sea estándar. Estos ficheros serán compilados por el sistema de corrección automática de la misma manera que se hace en el script autocorrector-plp2-1213.sh incluido en los programas de prueba, así que asegúrate de que tu práctica se compila y ejecuta adecuadamente con este script. Por último, el punto de entrada ha de estar incluido en una clase de nombre plp2:

```
class plp2 {
   public static void main(String[] args) throws Exception {
      plp2Lexer lex = new plp2Lexer(new ANTLRFileStream(args[0]));
      CommonTokenStream tokens = new CommonTokenStream(lex);
      plp2Parser parser = new plp2Parser(tokens);
      try {
         String trad = parser.s(); // S es la regla inicial
         System.out.println(trad);
      } catch (Exception e) {
         System.err.println(e.toString());
      }
   }
}
```

El archivo tgz ha de contener también un fichero denominado plp2.txt que incluya el DNI y nombre del autor de la práctica y el número de horas dedicadas a la realización de la práctica; todo esto en un formato como el siguiente:

```
AUTOR = 12345678; Pérez Pérez, Juan
HORAS = 16
```

Tu programa ha de funcionar sin problemas con la versión del compilador y de la máquina virtual del Java Development Kit 6 (JDK 6) instalada en los laboratorios y con la versión 3.4 de ANTLR. El fichero que contiene el autocorrector incluirá la versión adecuada de ANTLR, puedes (y deberías) utilizar ese fichero.

## Introducción

Esta práctica permite reciclar gran parte del trabajo realizado para la práctica 1, ya que tu traductor ha de realizar las misma tarea de traducción. Esta práctica, sin embargo, es más sencilla que aquella, ya que cuentas con la ayuda de la herramienta ANTLR.

En otras páginas de este wiki puedes encontrar más información sobre ANTLR y cómo utilizarlo.

## Especificación léxica y sintáctica

Los distintos componentes léxicos (*tokens*) del lenguaje fuente que han de ser reconocidos por el analizador léxico, así como los caracteres que pueden actuar de separadores y el formato de los comentarios, son iguales a los de la práctica 1.
Por otra parte, la gramática que describe el lenguaje fuente es la siguiente, que no puede ser modificada. Se trata de una gramática equivalente a la de la práctica 1, pero que utiliza notación EBNF. Esta notación permite trabajar con gramáticas más compactas que resultan más cómodas a la hora de traducir que la notación BNF. La notación EBNF es aceptada sin problemas por ANTLR.

S → **program id pyc** VSp  Bloque
VSp → (UnVsp)+
UnVsp → **function id dosp** Tipo **pyc** VSp Bloque **pyc**
UnVsp → **var** (V)+
V → **id** (**coma id**)* **dosp** Tipo **pyc**
Tipo → **integer**
Tipo → **real**
Bloque → **begin** SInstr **end**
SInstr → Instr (**pyc** Instr)*
Instr → Bloque
Instr → **id asig** E
Instr → **if** E **then** Instr (**else** Instr)? **endif**
Instr → **while** E **do** Instr
Instr → **writeln pari** E **pard**
E → Expr (**relop** Expr)?
Expr → Term (**addop** Term)*
Term → Factor (**mulop** Factor)*
Factor → **id**
Factor → **nentero**
Factor → **nreal**
Factor → **pari** Expr **pard**

### Mensajes de error léxico y sintáctico

Los mensajes de error léxico y sintáctico serán los que por defecto emite el código generado por ANTLR, salvo por  el hecho de que el análisis ha de detenerse tras la detección del primer error como se explica en esta página. Por tanto, no has de escribir código que procese estos mensajes (salvo por la redefinición de los métodos emitErrorMessage) y no se parecerán en muchos casos a los que emitía tu práctica 1.

En cualquier caso, dado que los errores emitidos por ANTLR son diferentes, por ejemplo, si el lexema de un token se especifica literalmente en una regla de la gramática o si se define un componente léxico para él, no se evaluarán los mensajes de error léxico o sintáctico en ninguno de los programas de prueba. Asegúrate, sin embargo, de que tu traductor emite errores léxicos y sintácticos aceptables, ya que pueden ser una buena fuente de información de cara a detectar algún fallo en tu especificación.

Para detectar una entrada no válida tras otra válida es conveniente añadir el token de ANTLR EOF tras la regla inicial, quedando algo así:
S :  **program id pyc** VSp  Bloque EOF

## Traducción

Toda la especificación semántica de la práctica 1, incluyendo los errores semánticos 5 al 12 (que no tienes que renumerar), es de aplicación en esta práctica.

## Planificación

- Comienza descargando el autocorrector, localizando el ANTLR y estudiando su funcionamiento.

- A continuación escribe y evalúa la especificación para ANTLR del analizador léxico y sintáctico de esta práctica.
- Una vez que el analizador léxico y el analizador sintáctico funcionen correctamente, diseña en papel el esquema de traducción dirigida por la sintaxis (ETDS). Este ETDS será muy similar al de la práctica 1, salvo por algunas ligeras modificaciones debidas a los cambios en la gramática por estar en notación EBNF.
- Ahora implementa cada una de las acciones semánticas en el fichero de ANTLR. Probablemente podrás aprovechar la mayoría de las clases auxiliares de la práctica 1.
- Evalúa finalmente tu traductor y ten en cuenta que cuanto más extensa sea la batería de programas de prueba que utilices, más oportunidades tendrás de salir airoso de la entrega de esta práctica.