**Interpreter**

**Definition**

Provides a definition of a macro language or syntax and parsing into objects in a program.

**Where to use & benefits**

* Need your own parser generator.
* Translate a specific expression.
* Handle a tree-related information.
* Related patterns include
  + [Composite](http://www.javacamp.org/designPattern/composite.html), which is an instance in an interpreter.
  + [Flyweight](http://www.javacamp.org/designPattern/flyweight.html), which shows how to share symbols with abstract context.
  + [Iterator](http://www.javacamp.org/designPattern/iterator.html), which is used to traverse the tree structure.
  + [Visitor](http://www.javacamp.org/designPattern/visitor.html), which is used to maintain behavior of each note in tree structure.

**Example**

Given any string expression and a token, filter out the information you want. The below is a simple parser program. the myParser method can be used to parse any expression. The composite, visit and iterator patterns have been used.

import java.util.\*;

class Parser{

private String expression;

private String token;

private List result;

private String interpreted;

public Parser(String e, String t) {

expression = e;

token = t;

}

public void myParser() {

StringTokenizer holder = new StringTokenizer(expression, token);

String[] toBeMatched = new String[holder.countTokens()];

int idx = 0;

while(holder.hasMoreTokens()) {

String item = holder.nextToken();

int start = item.indexOf(",");

if(start==0) {

item = item.substring(2);

}

toBeMatched[idx] = item;

idx ++;

}

result = Arrays.asList(toBeMatched);

}

public List getParseResult() {

return result;

}

public void interpret() {

StringBuffer buffer = new StringBuffer();

ListIterator list = result.listIterator();

while (list.hasNext()){

String token = (String)list.next();

if (token.equals("SFO")){

token = "San Francisco";

}else if(token.equals("CA")) {

token = "Canada";

}

//...

buffer.append(" " + token);

}

interpreted = buffer.toString();

}

public String getInterpretedResult() {

return interpreted;

}

public static void main(String[] args) {

String source = "dest='SFO',origin='CA',day='MON'";

String delimiter = "=,'";

Parser parser = new Parser(source, delimiter);

parser.myParser();

parser.interpret();

String result = parser.getInterpretedResult();

System.out.println(result);

}

}

java Parser

dest San Francisco origin Canada day MON