Atitit 词法分析器的设计最佳实践说明attilax总结

[1.1. 手写的优点：代码可读，对源代码中的各种错误给出友好的提示信息，用户体验高， 1](#_Toc10578)

[1.2. 使用状态表比较简单，dfa比较麻烦 1](#_Toc13202)

[1.3. 优先递归 替换循环 1](#_Toc28886)

[1.4. 内部分支switch替换if else 1](#_Toc8880)

[1.5. 单独的db que状态，内部可以容易包含single quo 1](#_Toc5924)

[1.6. 范例 1](#_Toc1780)

## 手写的优点：代码可读，对源代码中的各种错误给出友好的提示信息，用户体验高，

产品级别的编译器需要对源代码中的各种错误给出友好的提示信息。而DFA在这方面无法满足需要

## 使用状态表比较简单，dfa比较麻烦

## 优先递归 替换循环

递归可读性更好。。

## 内部分支switch替换if else

## 单独的db que状态，内部可以容易包含single quo

## 范例

/AtiPlatf\_ee/src/com/attilax/fsm/JavaTokenScannerV2.java

**public** **class** JavaTokenScannerV2 {

**public** **static** **void** **main**(String[] args) {

String **f** = pathx.*prjPath\_webrootMode*() + "/tokenT/a.txt";

String **s** = "meth(\\\"select from table where a='bb' \",'str2',\'s3\') ";

s = "meth(\"select from tab where a='abc'\",'str2','s3')";

// s = filex.read(f);

List<Token> **li** = **new** JavaTokenScannerV2().getTokens(s);

System.***out***.println(AtiJson.*toJson*(li));

}

**int** gColumn = -1;

@SuppressWarnings("unchecked")

**public** List<Token> **getTokens**(String codeStr) {

List<Token> **li** = Lists.*newArrayList*();

code\_char\_arr = codeStr.toCharArray();

**while** (**true**) {

Object **tk**;

**try** {

tk = nextTokens();

} **catch** (TokenEndEx **e**) {

**break**;

}

**if** (tk **instanceof** Token)

li.add((Token) tk);

**else** **if** (tk **instanceof** List)

li.addAll((Collection<? **extends** Token>) tk);

**else**

**throw** **new** RuntimeException("token type err,curchar:"+**this**.cur\_char+",colidx:"+**this**.gColumn);

}

**return** li;

}

**public** Object **nextTokens**() **throws** TokenEndEx {

// code\_char\_arr = code.toCharArray();

gColumn++;

**if** (gColumn > code\_char\_arr.length - 1)

**throw** **new** TokenEndEx(**new** String(code\_char\_arr));

cur\_char = code\_char\_arr[gColumn];

//cur\_char=cur\_char;

**if** (**this**.curTokenTxt.equals("1598"))

System.***out***.println("dbg");

**if**(**this**.gColumn==30)

System.***out***.println("dbg");

// get next char,,then change stat

// jude cur char and cur stat...then if or not chage stat

**switch**(cur\_char)

{

**case** '(':

**return** BrkStartEvt();

// break;

**case** ')':

**return** brkEndEvt();

**case** '\'':

**return** sQuoEvt();

**case** '\"':

**return** dbQuoEvt();

**case** ':':

**return** colonEvt();

**case** ',':

**return** commaEvt();

**default**:

**return** normalCharEvt();

//break;

}

**private** Object **normalCharEvt**() **throws** TokenEndEx {

// normal char

curTokenTxt = curTokenTxt + String.*valueOf*(cur\_char);

//gColumn += 1;

**return** nextTokens();

}

**rivate** Object **dbQuoEvt**() **throws** TokenEndEx {

// ---------------str type

// first dbquo

**if** ((cur\_char == '\"') && (**this**.curDbquoStat != "dbquoStart")) //

{

**this**.curStat = "dbquoStart";

**this**.curDbquoStat = "dbquoStart";

**this**.curTokenTxt = "";

**return** nextTokens();

}

// for close dbquo

**if** ((cur\_char == '\"') && **this**.curDbquoStat.equals("dbquoStart")) {

Token **tk3** = **new** Token(**this**.curTokenTxt).setType("str");

tokens.add(tk3);

**this**.curTokenTxt = "";

**this**.curStat = "dbquoEnd";

**this**.curDbquoStat = "dbquoEnd";

**return** tk3;

}

**return** normalCharEvt();

}