编译原理词法分析器说明文档

词法分析器功能

读取 java 代码类型的文本，对其中的内容进行词法分析，识别保留字，数字，字符串等，并输出 Token 序列，包括内容和类型。

分析器思路

1. 针对要识别的词给出 RE
2. 构造出每个 RE 对应的 NFA
3. 根据 NFA 转换成 DFA
4. 根据 DFA 进行代码实现

词语类型

保留字

类别为 Reserved Word，包括 void, class, public, private, protected, for, if, else,while, do, int, double, char, boolean, String, new, try, catch, static, return, this,main, switch, case, break, continue

标识符

类型为 Identifier

整数

类型为 Integer

小数

类型为 Double

操作符

类型为 Operator，包括+,++,+=,-,--,-=,*,*=,/,/=,|,||,&,&&,^,=,==,!,!=

标点

类型为 Punctuation，包括 . , ; : () [] {}

注释

类型为 Note，包含//型注释和/*…*/型注释

字符串

类型为 String

正则表达式

Identifier->letter (letter|digit)*
digit->0|1|2|3|4|5|6|7|8|9|0
letter->a|b|c|d|…|z
Integer->digit digit*
Double->Integer . digit digit*
Operator->+|++|=+|-|--|-=|*|*=| / |/=|=|==|&|&&| | | || |^| ! | !=
Punctuation->. | , | : | ; | (|) | [|] | { | }
ReservedWord-> void | class | public | private | protected | for | if | else |while | do | int | double | char | boolean | String | new | try | catch | static | return | this | main | switch | case | break | continue
Note->(// (letter|digit|punctuation|operator)*) | (/* (letter|digit|punctuation|operator)* */)
String->" (letter|digit|punctuation|operator)* "

输入输出示例

输入：

```
public boolean a;
a=true;
int b=5;
double c=1.2
/* This is note*/
b++;
String s="string";
c=(b+1)*2.5;
return c;
```

输出：

```
Token: public  Reserved Word
Token: boolean  Reserved Word
Token: a  Identifier
Token: ;  Punctuation
Token: a  Identifier
Token: =  Operator
Token: true  Identifier
Token: ;  Punctuation
Token: int  Reserved Word
Token: b  Identifier
Token: =  Operator
Token: 5  Integer
Token: ;  Punctuation
Token: double  Reserved Word
Token: c  Identifier
Token: =  Operator
Token: 1.2  Double
Token: /* This is note*/  Note
Token: b  Identifier
Token: ++  Operator
Token: ;  Punctuation
Token: String  Reserved Word
Token: s  Identifier
Token: =  Operator
Token: "string"  String
Token: ;  Punctuation
Token: c  Identifier
Token: =  Operator
Token: (  Operator
Token: b  Identifier
Token: +  Operator
Token: 1  Integer
Token: )  Operator
Token: *  Operator
Token: 2.5  Double
Token: ;  Punctuation
Token: return  Reserved Word
Token: c  Identifier
Token: ;  Punctuation
```