

Fasi di Compilazione Kitten

Fausto Spoto
fausto.spoto@univr.it

16 gennaio 2006

Un programma sorgente

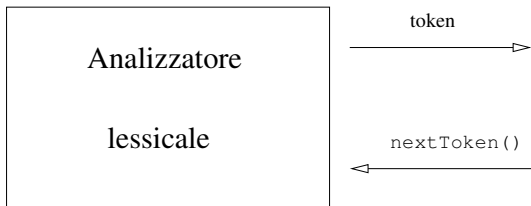
```
/* un misteriosissimo programma Kitten */  
  
class Fibonacci {  
    constructor() {}  
  
    method int fib(int n)  
        if (n = 0 | n = 1) then return 1  
        else return this.fib(n - 1) + this.fib(n - 2)  
}
```

Il risultato dell'analisi lessicale: make lexical

CLASS on 0-4
ID(Fibonacci) on 6-14
LBRACE on 16-16
CONSTRUCTOR on 20-30
LPAREN on 31-31
RPAREN on 32-32
LBRACE on 34-34
RBRACE on 35-35
METHOD on 40-45
INT on 47-49
ID(fib) on 51-53
LPAREN on 54-54
INT on 55-57
ID(n) on 59-59
RPAREN on 60-60

IF on 66-67
LPAREN on 69-69
ID(n) on 70-70
EQ on 72-72
INTEGER(0) on 74-74
OR on 76-76
ID(n) on 78-78
EQ on 80-80
INTEGER(1) on 82-82
RPAREN on 83-83
THEN on 85-88
RETURN on 90-95
INTEGER(1) on 97-97
ELSE on 103-106
...

Analisi lessicale on-demand



- Questo permette di effettuare l'analisi senza tenere in memoria l'intero testo del programma ma solo l'ultimo token
- Presuppone che la fase successiva sia capace di lavorare con un token alla volta

Inferenza dei tipi

I nodi dell'albero sintattico corrispondenti ad espressioni vengono decorati col loro *tipo statico* o di *dichiarazione*

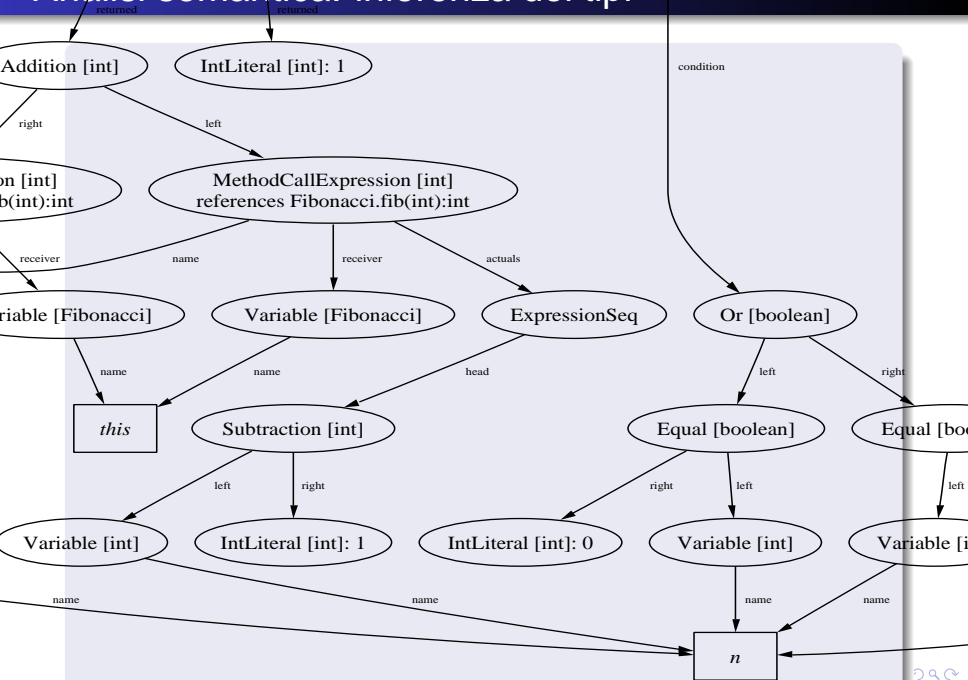
Controllo dei tipi

Ciascun nodo dell'albero sintattico viene controllato per vedere se i suoi figli hanno tipi che rispettano una qualche legge di *tipaggio statico*

Controlli aggiuntivi

- i `break` e `continue` devono occorrere solo all'interno di un ciclo
- un percorso di esecuzione in un metodo non `void` deve terminare con un'istruzione `return`
- i metodi ridefiniti in sottoclassi rispettano la controvarianza
- ...

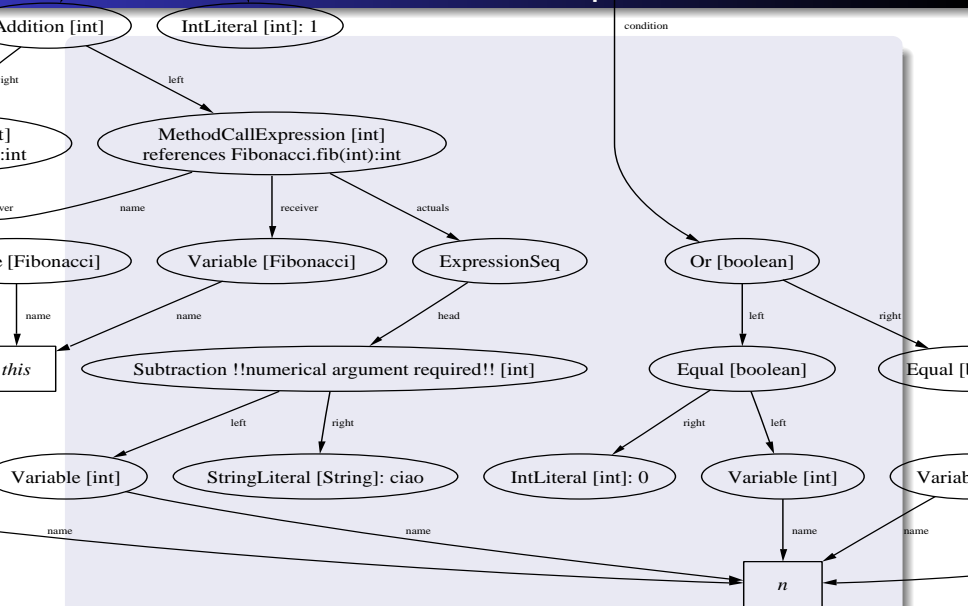
Analisi semantica: inferenza dei tipi



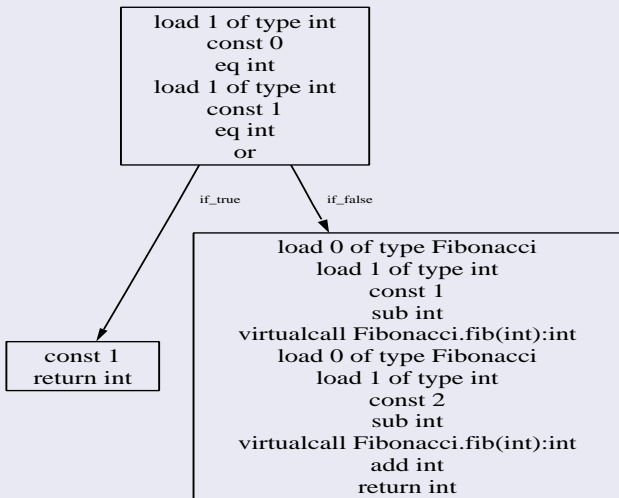
Introduciamo un errore

```
/* un misteriosissimo programma Kitten */  
  
class Fibonacci {  
    constructor() {}  
  
    method int fib(int n)  
        if (n = 0 | n = 1) then return 1  
        else return this.fib(n - "ciao")  
            + this.fib(n - 2)  
}
```

Analisi semantica: controllo dei tipi



La traduzione in codice intermedio Kitten bytecode: make translate



La traduzione in codice oggetto Java bytecode: makegeneratejb

```
0:  iload_1          21:  ifne           40
1:  iconst_0         24:  aload_0
2:  if_icmpeq 9     25:  iload_1
5:  iconst_0         26:  iconst_1
6:  goto 10          27:  isub
9:  iconst_1         28:  invokevirtual  fib:(I)I
10: iload_1          31:  aload_0
11: iconst_1         32:  iload_1
12: if_icmpeq 19    33:  iconst_2
15: iconst_0         34:  isub
16: goto 20          35:  invokevirtual  fib:(I)I
19: iconst_1         38:  iadd
20: ior              39:  ireturn
                        40:  iconst_1
                        41:  ireturn
```

Altre compilazioni possibili di Kitten

```
make clean
```

Ripulisce Kitten da tutti i file temporanei di emacs e dai `.class`

```
make javadocs
```

Rigenera la documentazione javadoc per il compilatore Kitten