

Fehlerbehandlung



- Panic Mode
 - Abbruch beim ersten Fehler
 - **Übung 3**
- Allgemeine Fangsymbole
 - Synchronisation der restlichen Eingabe mit der Grammatik
 - Parser kennt an jeder Stelle alle gültigen Nachfolge-Symbole
 - Aufwendig
- Spezielle Fangsymbole
 - Synchronisation nur an besonders "sicheren" Stellen.
 - Beispiele: Schlüsselwörter, Strichpunkte, ...
 - **Übung 4**



Beispiel: Deklarationen

Decl Part = { ForwardDecl } "{ Body }" .
ForwardDecl = "voi d" i dent "(" ")" ";" .
Body =

Damit lassen sich folgende Deklarationen erzeugen:

```
voi d p1 ();  
voi d p2 ();  
voi d p3 ();  
...  
{  
    ...  
}
```



Beispiel: Deklarationen

Decl Part = { ForwardDecl } "{" Body "}" .
ForwardDecl = "void" ident "(" ")" ";" .
Body =

```
private void DeclPart () {  
    while (sym == void_) {  
        ForwardDecl ();  
    }  
    check(lbrace); Body(); check(rbrace);  
}
```



Bsp: Fehler in *ForwardDecl*

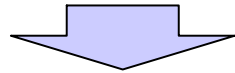
```
void p [ );  
{ ... }
```

	Erkenne Decl Part
next() → void_	Erkenne ForwardDecl
	void_ erkannt
next() → ident	ident erkannt
next() → lbrack	ERROR: "(" expected"
	ERROR: ")" expected"
	ERROR: ";" expected"
	ERROR: "{" expected"
	...
	ERROR: "}" expected"

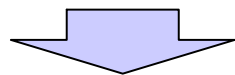


Bsp: First/Follow-Sets

Decl Part = { ForwardDecl } "{ " Body " }" .
ForwardDecl = "void" ident "(" ")" ";" .
Body =



First(ForwardDecl) = { void_ }
Follow(ForwardDecl) = First(ForwardDecl) + { lbrace } = { void_, lbrace }



```
private EnumSet<Token.Kind> followFwdDecl =  
    EnumSet.of(void_, lbrace, eof);
```



Beispiel: Deklarationen

Decl Part = { ForwardDecl } "{ Body }" .
ForwardDecl = "void" ident "(" ")" ";" .
Body =

```
private void DeclPart () {  
    for (;;) {  
        if (sym == void_) { ForwardDecl (); }  
        else if (followFwdDecl.contains(sym)) { break; }  
        else { recoverFwdDecl (); }  
    }  
    check(lbrace); Body(); check(rbrace);  
}
```

```
private void recoverFwdDecl () {  
    error("invalid forward declaration");  
    do {  
        scan();  
    } while (!followFwdDecl.contains(sym));  
}
```



Bsp: Fehler in *ForwardDecl* (2)

```
void p ();  
{ ... }
```

	Erkenne Decl Part
next() → void_	Erkenne ForwardDecl
	void_ erkannt
next() → ident	ident erkannt
next() → lbrack	ERROR: "(expected"
	ERROR: ") expected"
	ERROR: "; expected"
	ERROR: "invalid forward decl."
next() → rpar	
next() → semicolon	
next() → lbrace	lbrace erkannt
next() → ...	Erkenne Body
...	...
next() → rbrace	rbrace erkannt



LL(1)-Bedingung

- keine Alternativen mit gleichen terminalen Anfängen
- keine Linksrekursionen

➔ Bei Top-Down-Analyse:

mit einem Vorgriffssymbol entscheiden,
welche Alternative ausgewählt werden muss.

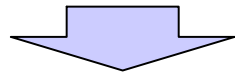
- Abhilfen:
 - gleiche Anfänge \Rightarrow Faktorisieren
 - Linksrekursionen \Rightarrow Umwandlung in Iteration



Regel *Statement*

```
Statement  
= Assignment  
| ProcedureCall  
| Increment | Decrement  
| ... .
```

gut lesbar, aber nicht LL(1), weil alle Alternativen mit `ident` beginnen



Abhilfe: Faktorisieren

```
Statement  
= Designator  
  ( Assignment Expr           // Assignment  
  | ActPars                 // ProcedureCall  
  | "++" | "--"             // Increment | Decrement  
  ) ";"  
| ... .
```

Beispiel: Kein LL(1)-Konflikt



$S = a B B B \mid b C.$ ($S = S_1 \mid S_2.$, $S_1 = aBBB.$, $S_2 = bC.$)
 $B = b B \mid a C.$ ($B = B_1 \mid B_2.$, $B_1 = bB.$, $B_2 = aC.$)
 $C = S S \mid c.$ ($C = C_1 \mid C_2.$, $C_1 = SS.$, $C_2 = c.$)

$\text{first}(S_1) \cap \text{first}(S_2) = \{a\} \cap \{b\} = \{\}$

$\text{first}(B_1) \cap \text{first}(B_2) = \{b\} \cap \{a\} = \{\}$

$\text{first}(C_1) \cap \text{first}(C_2) = \text{first}(S) \cap \{c\} = \{a, b\} \cap \{c\} = \{\}$

Beispiel: LL(1)-Konflikt



$S = a B B B \mid b C.$ ($S = S_1 \mid S_2.$ $S_1 = aBBB.$ $S_2 = bC.$)
 $B = b B \mid a C d.$ ($B = B_1 \mid B_2.$ $B_1 = bB.$ $B_2 = aCd.$)
 $C = [S S \mid c].$ ($C = C_1 \mid C_2 \mid C_3.$ $C_1 = SS.$ $C_2 = c.$ $C_3 = \epsilon.$)

$FC1 = \text{first}(C_1) = \text{first}(S) = \{a, b\}$
 $FC2 = \text{first}(C_2) = \{c\}$
 $FC3 = \text{first}(C_3) = \text{follow}(C) =$
 $= \{d\} \cup \text{follow}(S) =$
 $= \{d\} \cup \text{first}(S) \cup \text{follow}(C) =$
 $= \{d\} \cup \{a, b\} =$
 $= \{a, b, d\}$

$FC1 \cap FC2 = \{\}$
 $FC2 \cap FC3 = \{\}$
 $FC1 \cap FC3 = \{a, b\}$

Beispiel: LL(1)-Konflikt



S = a B B B | b C.
B = b B | a C d.
C = [S S | c].

Beispielsatz: a a b b d a d a d

S = a B B B
B = a C d
C = S S
S = b C
C = S ...



UE 3: Syntaxanalyse (*Parser*)

- Keine neuen Angabe- und Test-Klassen
- Abgabe
 - siehe Abgabeanleitung auf Homepage!
 - elektronisch bis Mi, 12.11.2008, 20:15
 - alle Java-Dateien des Compilers
 - keine class-Dateien
 - keine Testfälle
 - auf Verzeichnisstruktur achten
 - auf Papier
 - nur Parser.java