

# Struktur eines Compilers

---

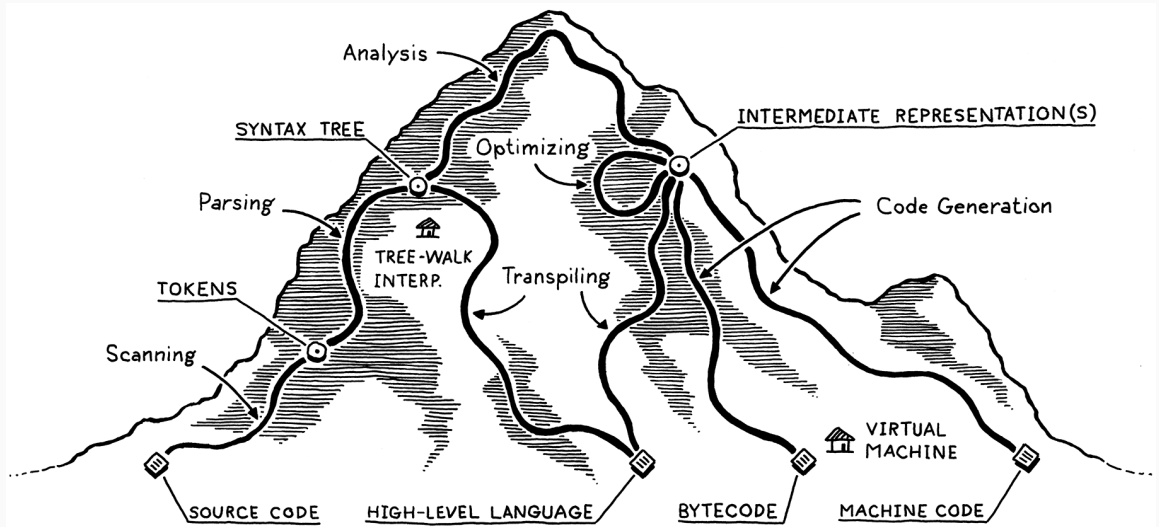
Carsten Gips (HSBI)

Unless otherwise noted, this work is licensed under CC BY-SA 4.0.

*The cat runs quickly.*

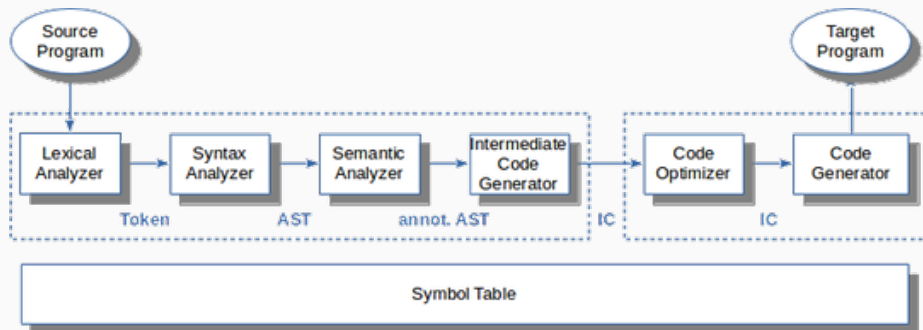
=> Struktur? Bedeutung?

# Compiler: Big Picture



Quelle: A Map of the Territory (mountain.png) by Bob Nystrom on Github.com (MIT)

# Compiler: Prinzipieller Aufbau



## Lexikalische Analyse: Wörter (“*Token*”) erkennen

```
sp = 100;
```

## Lexikalische Analyse: Wörter (“*Token*”) erkennen

```
sp = 100;
```

```
<ID, sp>, <OP, =>, <INT, 100>, <SEM>
```

# Syntaxanalyse: Sätze erkennen

<ID, sp>, <OP, =>, <INT, 100>, <SEM>

# Syntaxanalyse: Sätze erkennen

<ID, sp>, <OP, =>, <INT, 100>, <SEM>

```
statement : assign SEM ;  
assign : ID OP INT ;
```



# Syntaxanalyse: Sätze erkennen

$\langle \text{ID}, \text{sp} \rangle, \langle \text{OP}, = \rangle, \langle \text{INT}, 100 \rangle, \langle \text{SEM} \rangle$

statement : assign SEM ;

assign : ID OP INT ;

```
      statement
      /      \
    assign    SEM
   /  |  \    |
  ID  OP INT  ;
  |   |   |
  sp  =  100
```

```
      =
     /  \
    sp   100
```

## Vorschau: Parser implementieren

```
stat : assign | ifstat | ... ;  
assign : ID '=' expr ';' ;
```

```
void stat() {  
    switch (<<current token>>) {  
        case ID : assign(); break;  
        case IF : ifstat(); break;  
        ...  
        default : <<raise exception>>  
    }  
}  
  
void assign() {  
    match(ID);  
    match('=');  
    expr();  
    match(';');  
}
```

# Semantische Analyse: Bedeutung erkennen

```
{  
  int x = 42;  
  {  
    int x = 7;  
    x += 3;    // ???  
  }  
}
```

# Semantische Analyse: Bedeutung erkennen

```
{  
  int x = 42;  
  {  
    int x = 7;  
    x += 3;    // ???  
  }  
}
```

sp = 100;

```
      = {type: real, loc: tmp1}  
      / \  
     /   \  
    sp   inttofloat  
 {type: real,    |  
   loc: var b}  100
```

## Zwischencode generieren

```
      = {type: real, loc: tmp1}
    /  \
  /      \
sp      inttofloat
{type: real,   |
 loc: var b}   100
```

=> `t1 = inttofloat(100)`

`t1 = inttofloat(100)`  $\Rightarrow$  `t1 = 100.0`

`x = y*0;`  $\Rightarrow$  `x = 0;`

- Maschinencode:

```
STD  t1, 100.0
```

- Andere Sprache:

- Bytecode
- C
- ...

# Probleme

$5*4+3$

AST?



# Probleme

5\*4+3

AST?

```
stat : expr ';'
      | ID '(' ' ' ')' ' ';
;

expr : ID '(' ' ' ')'
      | INT
      ;
```

- Compiler übersetzen Text in ein anderes Format
- Typische Phasen:
  1. Lexikalische Analyse
  2. Syntaxanalyse
  3. Semantische Analyse
  4. Generierung von Zwischencode
  5. Optimierung des (Zwischen-) Codes
  6. Codegenerierung



Unless otherwise noted, this work is licensed under CC BY-SA 4.0.

## Exceptions

- Figure A Map of the Territory (mountain.png) by Bob Nystrom on Github.com (MIT)