# Die Sprachstandards

Lars Thoms

Universität Hamburg

Proseminar »C – Grundlagen und Konzepte«, 2013

# Inhalt

# Was ist ein Sprachstandard?

- Beschreibt die Grammatik/Syntax einer Sprache
- Sie vereinheitlicht die verschiedenen Sprachstile

Standardaufbau

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
   printf("Hello World!");
   exit(EXIT_SUCCESS);
}
```

# Welche Sprachstandards gibt es?

- C89 und C90 von ANSI
- C95 und C99 von ISO
- C11 ist »bald« fertig
- Neue Standards erweitern bestehende Standards

# Syntax

- Statements werden mit Semikolon geschlossen
- Blöcke werden mit geschweiften Klammern eingeschlossen

```
foo(arg1, arg2, ...);

if( Bedingungen ) { ... }

switch( Bedingungen )
{
  case Wert:
    ...
  default:
    ...
}
```

# Variablen

Eigenschaften

- veränderlich
- liegen auf dem Stack bzw. auf dem Heap

Deklaration

- C89: am Anfang eines Blocks
- ab C99: muss nicht am Anfang eines Blocks deklariert werden

```
type name = value;

int  foo  = 42;
```

# Konstanten

Eigenschaften

- unveränderlich
- wird direkt im Programmcode eingebettet

Deklaration

```
const type name = value;

const int   foo  = 42;
```

# Primitive Datentypen

Ganze Zahlen

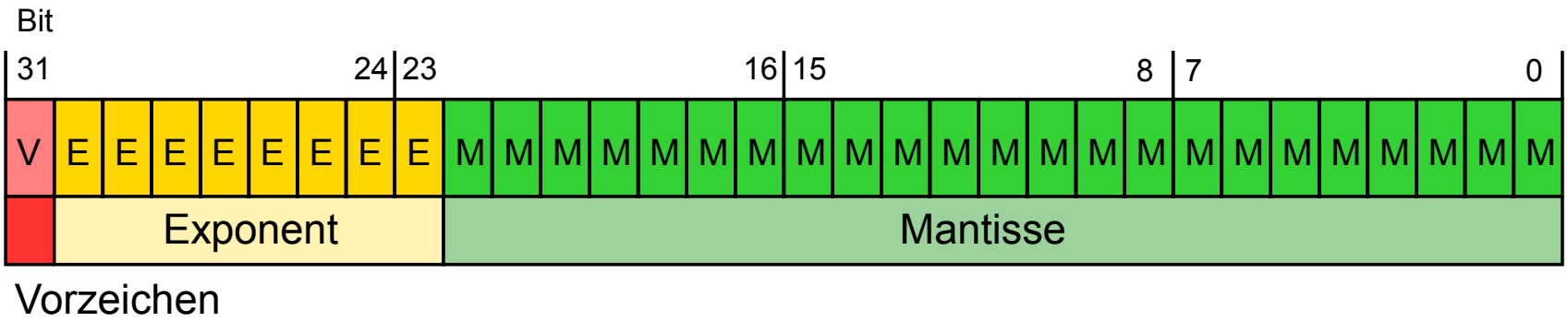
- *int* muss mindestens 2 Byte und *char* genau 1 Byte breit sein
- *signed char* $\leq$ *short int* $\leq$ *int* $\leq$ *long int* $\leq$ *long long int*

| Modell | char | short | int | long | long long | Plattform |
|---|---|---|---|---|---|---|
| IP16 | 8 | 16 | 16 | 32 | 64 | MS DOS im SMALL memory model |
| LP32 | 8 | 16 | 16 | 32 | 64 | MS DOS im LARGE memory model |
| ILP32 | 8 | 16 | 32 | 32 | 64 | x86-System |
| LLP64 | 8 | 16 | 32 | 32 | 64 | Windows x86-64 |
| LP64 | 8 | 16 | 32 | 64 | 64 | Unix/Linux x86-64 |
| ILP64 | 8 | 16 | 64 | 64 | 64 | SPARC64 |

# Primitive Datentypen

Gleitkommazahlen

- *float*, *double* und *long double*
- meistens im *IEEE 754*-Format




```
float foo = 1.0;
float bar = 1f;
```

# Primitive Datentypen

Komplexe Zahlen

- ab C99
- *float complex*, *double complex* und *long double complex* via *complex.h*
- nicht auf allen Umgebungen verfügbar

Logische Typen

- ab C99
- *bool* via *stdbool.h*

# Arrays

## Char-Array / Zeichenkette

```
char foo[12] = "Hello World!";
```

## Multidimensionales Array

```
int foo[2][3] = {{13,3,7},{0,4,2}};

foo[0][0] = 1;
foo[0][1] = 33;

printf("%d", foo[1][2]); //Output: 2
```

# Aufzählung »enum«

```
enum type_name
{
   ...
} name;
```

```
enum week
{
  Mon=1, Tue, Wed, Thu, Fri, Sat, Sun
};

printf("%d", Fri); //Output: 5
```

# Struktur »struct«

```
struct type_name
{
  ...
} name;
```

```
struct Foo
{
  int a, b, c;
};

struct Foo bar;

bar.a = 42;
```

# Typumwandlung

```c
float a = 10.0;

int b = (int) a;

printf("%d", b); //Output: 10
```

# Arithmetische Operatoren

- `~` Bitweises NICHT
- `*` Multiplikation
- `/` Division
- `%` Modulo
- `+` Addition
- `-` Subtraktion
- `«` Bitweises shiften
- `&` Bitweises UND
- `^` Bitweises XOR
- `|` Bitweises ODER

```
a = b % c;
```

# Zuweisungsoperatoren

## Direkte Zuweisung

```
a = b + c;
```

## Kombinierte Zuweisungen

```
a = a + b;   <=>   a += b;

a = b = c;

a <<= 2     //Shift

foo(++a); //Praefix-Inkrement
foo(a++); //Postfix-Inkrement
```

# Vergleichsoperatoren

| | |
|---|---|
| `<` | Kleiner |
| `>` | Größer |
| `<=` | Kleiner gleich |
| `>=` | Größer gleich |
| `==` | Gleichheit |
| `!=` | Ungleichheit |

```
if (a == b) { ... }
```

# Aussagenlogik

| | |
|---|---|
| ! | Logisches NICHT |
| && | Logisches UND |
| \|\| | Logisches ODER |

```
if((a == b) || (!a == c)) { ... }
```

# Deklaration

```
return_type name(type param1, ...)
{
   ...

   return param1;
}
```

# Beispielfunktion

```
void print_usage(int error_type)
{
  switch(error_type)
  {
    case 1 :
      printf("Too many arguments");
      exit(EXIT_FAILURE);
    case 2 :
      printf("File not found");
      exit(EXIT_FAILURE);
    default:
      printf("Unknown error");
      exit(EXIT_FAILURE);
  }
}
```

# Einstiegsfunktion

```
$ ./program.x foobar
```

```
int main(int argc, char *argv[])
{
   ...

   printf("%s", argv[0]); //Output: ./program.x

   printf("%s", argv[1]); //Output: foobar

   exit(EXIT_SUCCESS);
}
```

# Header

- praktisch zur Modularisierung von Software
- enthält Funktionsprototypen und Definitionen
- Deklarationen befinden sich nicht im Header

demo.h

```
extern int foo;      //Definition
int bar(int);
```

demo.c

```
#include "demo.h"
int foo = 42;        //Deklaration
int bar(int a) { ... }
```

# Positionen von Header

Eigene Header

```
#include "demo.h"
#include "foo/bar.h"
```

Systemheader (z.B. */usr/include/**)

```
#include <stdlib.h>
#include <stdio.h>
```

# Standardheader

- Standardbibliothek: *stdlib.h*
- Ein-/Ausgabe: *stdio.h*
- Strings: *string.h*
- Boolesche Werte: *stdbool.h*

# Ausgabe

- es gibt zwei Ausgabearten:
  - stdout (Standardausgabe)
  - stderr (Fehlerausgabe)

```
fprintf(stdout, "Hello World!"); //Standardausgabe

fprintf(stderr, "Goodbye World!"); //Fehlerausgabe
```

# Eingabe

- Eingabe via Terminal
- Speicherüberläufe möglich
- Anzahl der Einlesesymbole begrenzen!!!

```
char foo[42];

fscanf(stdin, "%42s", &foo);
```

# Neuerungen in C11

- Thread-Unterstützung
- Bessere Unterstützung von UTF-8/16/32
- Transparente Structs
- Exklusiver Dateizugriff
- Abwandlung von Überladung

# Zusammenfassung

- Sprachstandards sind wichtig, da sie Grammatik und Syntax für die Sprache festlegen

$\rightarrow$ Kompatibilität wahren

- Ein neuer Sprachstandard ist niemals destruktiv!

# Quellen


- http://www.heise.de/developer/artikel/C11-Neue-Version-der-Programmiersprache-Teil-1-1661014.html
- http://www.open-std.org/jtc1/sc22/wg14/www/standards
- http://upload.wikimedia.org/wikipedia/commons/5/56/IEEE-754-single.svg
- http://de.wikipedia.org/wiki/C_%28Programmiersprache
- http://upload.wikimedia.org/wikipedia/de/9/91/Logo_der_ISO.svg
- http://www.smart-v2g.info/images/logos/iec-logo.svg
- http://de.wikipedia.org/wiki/C_%28Programmiersprache%29#Datenmodell
- http://en.wikipedia.org/wiki/C99#Design
- http://en.wikipedia.org/wiki/C11_%28C_standard_revision%29
- http://de.wikibooks.org/wiki/C-Programmierung:_Ausdr%C3%BCcke_und_Operatoren