Programmieren in C

mehr Herumgezeige

thoto

/dev/tal e.V.

13. April 2013 (Version vom 22. Mai 2014)

Agenda für Heute

1. Vertiefung
   - Wahrheitswerte
   - Makefile
2. Zeiger Teil II
   - Wiederholung
   - Speicherreservierung
   - Zeichenketten

Wahrheitswerte

Die Wahrheit

- Wahrheitswerte bei if, case, ...

Die Wahrheit

- Wahrheitswerte bei if, case, ...
- Was ist wahr?

Die Wahrheit

- Wahrheitswerte bei if, case, ...
- Was ist wahr?
- 0 $\Rightarrow$ Falsch

Die Wahrheit

- Wahrheitswerte bei if, case, ...
- Was ist wahr?
- 0 $\Rightarrow$ Falsch
- Was nicht falsch ist wahr.

Die Wahrheit

- Wahrheitswerte bei if, case, ...
- Was ist wahr?
- 0 $\Rightarrow$ Falsch
- Was nicht falsch ist wahr.
- „Boolean“ existiert nicht.

Binäroperatoren

AND && (logisches Und)

OR || (logisches Oder)

NOT ! (logisches Nicht)

Binäroperatoren

AND	&& (logisches Und)
OR	|| (logisches Oder)
NOT	! (logisches Nicht)
Gleich	==
Ungleich	!=
	Größergleich >= Kleinergleich <=
	Größer > Kleiner <

zum Thema Makefile ...

- verschieden Ziele

zum Thema Makefile ...

- verschieden Ziele
- nützlich: clean

zum Thema Makefile ...

- verschieden Ziele
- nützlich: clean

clean

```
clean:
    rm datei datei.o
```

noch mehr Gezeige

Copyright: mkorsakov/flickr

noch einmal: Referenzierung und Dereferenzierung

- Bezeichnung für Häuser: Supermarkt, Post, Bank ...

noch einmal: Referenzierung und Dereferenzierung

- Bezeichnung für Häuser: Supermarkt, Post, Bank ...
- Nicht ausreichend für große Städte

noch einmal: Referenzierung und Dereferenzierung

- Bezeichnung für Häuser: Supermarkt, Post, Bank ...
- Nicht ausreichend für große Städte
- daher Hausnummern und Straßen

Dynamisch Speicher holen

```
foo=malloc(1024); //reserviert 1024 Byte
foo=malloc(12); //reserviert 12 Byte
free(foo); //gibt die 12B wieder frei, aber nicht d
```

Achtung: Rückgabewerte

Rückgabewerte

VORSICHT bei Rückgabwerten:
Speicher bereits freigegeben nach Funktionsende! malloc() und free() nutzen!

Speicherlecks

Vorsicht bei Speicherlecks

Nicht auf Sachen außerhalb des reservierten Bereiches zugreifen

Beispiel: Zeichenkette mit Zeigern

```
char hello[] = "Hello World!\n";
char* z=hello; //Zeiger ist Array!
while((*z)!=0x00){
   putchar(*z);
   z++;
}
```

Stringfunktionen

- `#include <string.h>`

Stringfunktionen

- `#include <string.h>`
- `size_t strlen(const char *s);`
- `char *strcat(char *dest, const char *src);`
- `char *strncat(char *dest, const char *src, size_t n);`
- `char *strstr(const char *haystack, const char *needle)`

Stringfunktionen

- `#include <string.h>`
- `size_t strlen(const char *s);`
- `char *strcat(char *dest, const char *src);`
- `char *strncat(char *dest, const char *src, size_t n);`
- `char *strstr(const char *haystack, const char *needle)`
- ...