

Programmierwerkzeuge unter Unix

Tutorium der Unix-AG

Jean-Marie Gaillourdet

Christian Schmidt

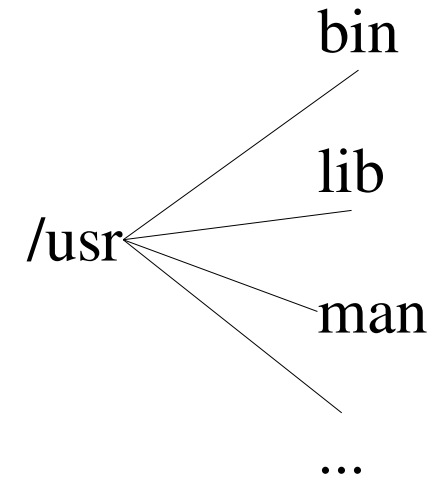
Mathias Dalheimer

www.unix-ag.uni-kl.de

- Übersetzung fremder Software
- Übersetzung eigener Software
- Übersetzungswerkzeuge

- Software wird als Quellcode erstellt
 - In Java, C, C++, etc.
 - Compiler übersetzt in Binärcode für CPU und Betriebssystem
 - Linker verbindet Programmteile
 - Installation verteilt die Programmteile damit sie gefunden werden
- viele Abhängigkeiten vom Betriebssystem

- Hierarchisches Dateisystem
- Suchpfade: Programme, Bibliotheken, Dokumentation, etc.



- Alternative Positionen:
 - `/usr/local` oder auch im Home: `~j_gaillo/install`

- Quellcode kopieren

```
cp ~j_gaillo/units-1.55.tar.gz .
```

- Konfigurieren

```
./configure --prefix=${HOME}/install
```

- Übersetzen

```
make
```

- Installieren

```
make install
```

Übersetzung eigener Software



- Linken kann auch beim Programmstart passieren

1. Beispiel in C

```
#include <stdio.h>

int main( int argc, char** argv ) {
    printf( "Hello World\n" );
}
```

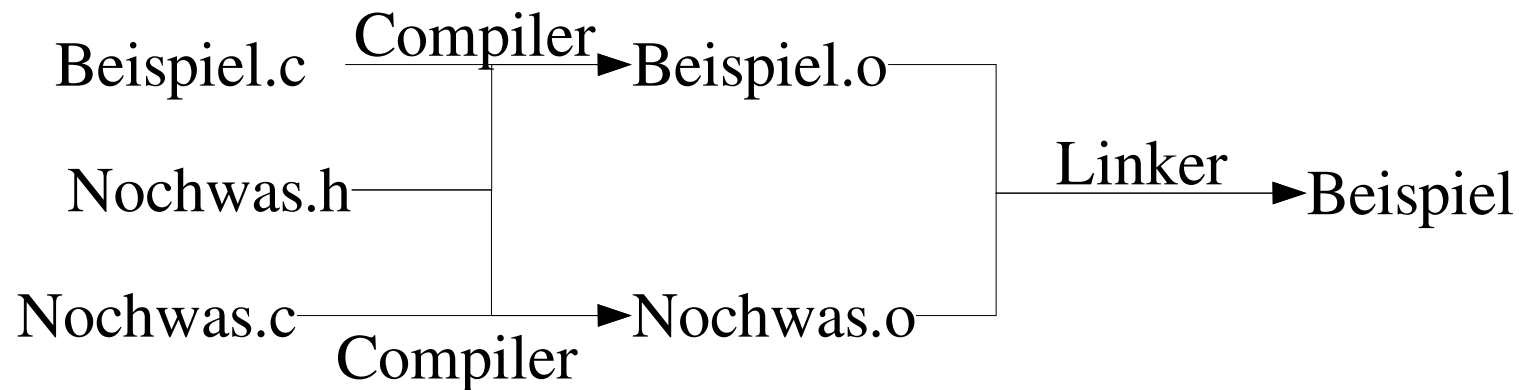
```
gcc -c HelloWorld.c
```

```
gcc -o HelloWorld HelloWorld.o
```

```
./HelloWorld
```

2. Beispiel in C

- 3 Dateien: Beispiel.c Nochwas.c Nochwas.h



```
gcc -c Beispiel.c
```

```
gcc -c Nochwas.c
```

```
gcc Nochwas.o Beispiel.o -o Beispiel
```

- Quellcode in der Datei
`unixag/SwingApplication.java`
- Übersetzung
`javac unixag/SwingApplication.java`
- Start
`java unixag.SwingApplication`
- Package `unixag`

- HelloWorld übersetzen und ausführen
- 2. C-Beispiel übersetzen und ausführen
- Java Beispiel übersetzen und ausführen

Übersetzungswerkzeuge

- Überblick zu behalten ist schwer
 - Was muss jetzt neu übersetzt werden?
- Große Programme bestehen aus mehreren hundert Dateien
- Automatisierung wünschenswert

Automatisierung mit Skripten

- Alle Befehle untereinander in ein Shellskript schreiben
- Einfach
- Alle Dateien werden immer übersetzt
 - langsam

Automatisierung mit make

- Automatische Aktualisierung nach Änderungen
- Anwendungen:
 - Programmieren
 - LaTeX
- Make erkennt Abhängigkeiten und führt alle „nötigen“ Befehle aus

http://www.delorie.com/gnu/docs/make/make_toc.html

Beispiel.dvi: Beispiel.tex

■ latex Beispiel.tex

Beispiel.ps: Beispiel.dvi

■ dvips -o Beispiel.ps Beispiel.dvi

view: Beispiel.dvi

■ xdvi Beispiel.dvi

- Testet mal die einzelnen Targets des Makefiles.
- Öffnet den LaTeX-Quelltext in einem Editor und verändert etwas am Text.
- Danach wieder make view aufrufen. Was passiert dann?

Nächste Woche: Texteditoren mit Christian

Vielen Dank für Eure Aufmerksamkeit