Debuggen auf der Standardausgabe

Autor: René Doß info@dossmatik.de
Datum: 3.8.2010

Einfache Ausgaben sind häufig hilfreich, um interne Zustände des Programms zu verfolgen. Die Fehlersuche oder der Nachweis der korrekten Abarbeitung kann häufig durch eine Ausgabe vereinfacht werden. Die einfachste Variante ist die zu Hilfenahme der Standardausgabe. Der Befehl printf() gibt den Wert einer Variablen aus und kann so zum Debuggen genutzt werden.

Bsp.

```
printf(„Meldung Variable int a: %d“,a);
```

Sobald das Programm läuft und diese Zeile abgearbeitet wird, wird der Wert von a auf der Standardausgabe ausgegeben.

Diese Ausgabe lässt sich mit zusätzlichen Informationen ergänzen. Sehr hilfreich sind Präprozessor Anweisungen. Hier für gibt es spezielle Platzhalter, die der Präprozessor gegen den entsprechenden Wert vor dem Kompilieren ersetzt.

`__FILE__`	der Filenamen als String
`__FUNCTION__`	Die Funktion in dem sich der Output befindet als String
`__LINE__`	Zeilennummer als int

Diese speziellen Platzhalter können die Fehlerausgabe mehr Inhalt ausgeben und so schneller der Fehler gefunden werden. Auf Dauer werden die vielen Ausgaben auf dem Bildschirm lästig und man kommentiert alle Ausgaben aus, bis auf diejenigen die gerade untersucht werden. Das ganze hat einen Nachteil man ist ständig am um kommentieren und am neu kompilieren. Hierfür ist eine sinnvoll eine Möglichkeit des an- abschalten der Debugmeldungen. Das kann durch einen Parameter bei Programmaufruf erfolgen.

```
/*********************************************
debug_example.c     written by R.Doss
www.dossmatik.de
*********************************************/
#include <stdio.h>


#define ERROR(fmt, args...) fprintf(stderr, "ERROR - %s,%s(): " fmt,
__FILE__, __FUNCTION__, ## args)
#define DEBUG(fmt, args...) {if (debug_mode) fprintf(stderr, "%s:%d "
fmt, __FILE__,__LINE__, ## args);}


int debug_mode=0;
```

```
int function_demo(void)
{
      DEBUG(" just arrived \n");
      ERROR("Hier ist es falsch\n");
return -1;
}

int main (int argc, char* argv[])
{
if (argc > 1){
      if ( (strcmp(argv[1], "-d") == 0)) debug_mode = 1;
      }

function_demo();

   DEBUG("Debug %d\n",debug_mode);

   return 0;
}
```

Ein Programmaufruf bringt folgenden Output.

```
red>./debug_example
ERROR - debug_example.c,function_demo(): Hier ist es falsch

red> ./debug_example -d
debug_example.c:14  just arrived
ERROR - debug_example.c,function_demo(): Hier ist es falsch
debug_example.c:29 Debug 1
```

Die Makros Debug und Error erzeugen ein sehr nützliches Verhalten für die Fehlersuche. Das Makro Debug ist an- und abschaltbar mit dem zweiten Parameter bei dem Programmaufruf. Es gibt den Quellcodedateinamen und die Zeile der Datei aus. Bei Programmen, die aus mehreren Quellcodedateien bestehen, ist diese Information wichtig. Die hier vorgestellten Makros Debug() und Error() erzeugen ein sehr nützliches Verhalten für die Fehlersuche. Mit den Präprozessor Anweisungen kann man sehr effektiv arbeiten.