Inhaltsverzeichnis

1. Übung: LOGGING

1.1. Aufgabe: G2LOG erstellen

1.1.1. First: Download, unzip and build

1. Download: https://bitbucket.org/KjellKod/g2log/downloads
2. **unzip:**
   if on unix: unzip into an unix filesystem (not vFat, ...)
3. build: (on Linux)
   cd g2log
   mkdir build
   cd build
   cmake ..
   make

For other OS see:

```
# WINDOWS == README: Example how to setup environment + running an example
# Below written for VS11 (2012)
# 1. please use the "Visual Studio Command Prompt 11 (2012)"
# 2. from the g2log folder
#    mkdir build
#    cd build;
# 3. cmake -DCMAKE_BUILD_TYPE=Release -G "Visual Studio 11" ..
#               the "Visual Studio 11" .. does not require just::thread!
# 4. msbuild g2log_by_kjellkod.sln /p:Configuration=Release
# 5. Release\g2log-FATAL-example.exe
#
# . LINUX:To try this out from folder g2log:
#    mkdir build
#    cd build
#    cmake ..    # create makefiles in g2log/build directory
#    make    # link active_object, g2log and example code to get an "example"
executable
#     ./g2log-FATAL-example
#
#   Clang on Linux
#   From g2log/g2log
#   mkdir build && cd build
#   cmake -DCMAKE_CXX_COMPILER=clang++ ..
#   You can use:   "VERBOSE=1 make"  if you want to double-check settings
otherwise just run
```

```
#   "make -j"
```

1.1.2. Run Test example

1. `cd g2log/build`
2. `./g2log-FATAL-example`
3. `see: cat /tmp/g2log-FATAL-example.g2log.20150217-134352.log`

1.1.3. compile and run your own main.cpp

1. `cd g2log/test_example`
2. `g++ -std=c++11 -I ../src main.cpp -L../build -llib_g2logger -lpthread`

`-L` .... Library Path
`-l` .... library name lib_g2logger (is different from filename which is liblib_g2logger.a

`-I` .... Path of header files
`-std=c++11` .... use c++11 Version

1.2. Aufgabe: Syslog aus der glibc verwenden

Erstellen Sie eine kurze Zusammenfassung inkl. einem kleinen Beispiel.

http://www.infodrom.org/~joey/Writing/Linux-Magazin/syslog.html

http://www.linuxselfhelp.com/gnu/glibc/html_chapter/libc_18.html

```
/*
http://www.gnu.org/software/libc/manual/html_node/Syslog-Example.html

description:
18.2.5 Syslog Example

Here is an example of openlog, syslog, and closelog:

This example sets the logmask so that debug and informational messages
get
discarded without ever reaching Syslog. So the second syslog in the
example does nothing.

*/

#include <syslog.h>

int main(){
     setlogmask (LOG_UPTO (LOG_NOTICE));

     openlog ("exampleprog", LOG_CONS | LOG_PID | LOG_NDELAY,
LOG_LOCAL1);
```

```
     syslog (LOG_NOTICE, "Program started by User %d", getuid ());
     syslog (LOG_INFO, "A tree falls in a forest");

     closelog ();

     return 0;
}




/*
## RUN:
     shell 1:
          sudo xconsole -file /dev/xconsole
     shell 2:
          logger "hello world!!!"
     shell 3:
          ./test-syslog.exe



## SEE ALL LOGS:
     sudo cat /var/log/messages


## configure on debian
cat /etc/rsyslog.conf
```

1.3. +Aufgabe: libg2log.a erstellen (old version)

https://bitbucket.org/KjellKod/g2log/src

Wir wollen die Library libg2log.a erstellen und testen:

1. Entpacken Sie die Datei: g2log.zip und
2. studieren Sie die beiden Makefile und
3. studieren Sie das Programm: main.cpp
4. make lib
5. make main
6. make run
7. Vergleichen Sie den Inhalt der LOG-Datei mit den Anweisungen in main.cpp

Wir wollen nun die Division durch 0 testen:

8. Ändern Sie main.cpp:

    1. // **CHECK(2<0)** << "TEST CHECK(): to see if design by contract works." <<
    2. "This should be the LAST LINE of the log, and the programm will be terminated";
    3.
    4. // try this
    5. **int x= 1000 / 0;**

9. make main
10. make run

11. Vergleichen Sie den Inhalt der LOG-Datei mit den Anweisungen in main.cpp

1.4. Fragen

Beantworten Sie die folgenden Fragen:

1. **Singleton-muster**
   1. Welche Methoden müssen private gehalten werden? Und warum?

2. **g2log**
   1. Welche Logging-level gibt es?
   2. In welcher Datei sind diese definiert?
   3. Beschreiben Sie in aller Kürze, was g2log alles kann.
   4. Was bedeuten die Flags -L und -l beim Übersetzen mit g++ ?