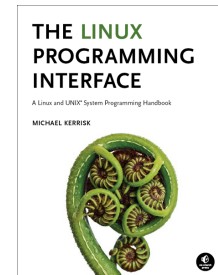


Building and Using Shared Libraries on Linux

Course code: M7D-SHLIB01



This course provides a thorough understanding of the process of designing, building, and using shared libraries on Linux. Detailed presentations coupled with carefully designed practical exercises provide participants with the knowledge needed to understand, design, create, and deploy shared libraries.

Audience and prerequisites

The primary audience comprises designers and programmers building and deploying shared libraries on Linux. Systems administrators are likely to also find the course of benefit for the purpose of troubleshooting problems with shared libraries.

Participants should have a good reading knowledge of the C programming language and some programming experience in a language suitable for completing the course exercises (e.g., C, C++). No previous experience of working with shared libraries is required.

Course materials

- A course book (written by the trainer) that includes all course slides and exercises
- An electronic copy of the trainer's book, *The Linux Programming Interface*
- A source code tarball containing all of the example pro-

grams written by the trainer to accompany the presentation

Course duration and format

One day, with around 50% devoted to practical sessions.

Course inquiries and bookings

For inquiries about courses and consulting, you can contact us in the following ways:

- Email: training@man7.org
- Phone: +49 (89) 2155 2990 (German landline)

Prices and further details

For course prices, upcoming course dates, and further information about the course, please visit the course web page, <http://man7.org/training/shlib/>.

About the trainer



Michael Kerrisk has a unique set of qualifications and experience that ensure that course participants receive training of a very high standard:

- He has been programming on UNIX systems since 1987 and began teaching UNIX system programming courses in 1989.
- He is the author of *The Linux Programming Interface*, a 1550-page book widely acclaimed as the definitive work on Linux

system programming.

- He is actively involved in Linux development, working with kernel developers on testing, review, and design of new Linux kernel-user-space APIs.
- Since 2004, he has been the maintainer of the Linux *man-pages* project, which provides the manual pages documenting the Linux kernel-user-space and GNU C library APIs.

Building and Using Shared Libraries on Linux: course contents in detail

1. Basics of shared library creation and use

- Static versus shared libraries
- Creating a shared library
- The dynamic linker
- Using a shared library
- `LD_LIBRARY_PATH`
- The library *soname*

2. Shared library versions and naming conventions

- Requirements of a shared library versioning scheme
- Library real names, sonames, and linker names
- Creating a shared library following standard conventions
- Installing a shared library
- *ldconfig*

3. Symbol resolution

- Impact of library load order on symbol resolution
- Link maps (“namespaces”)

4. Finding shared libraries at run time

- Rpath lists (`DT_RPATH` and `DT_RUNPATH`)
- How the dynamic linker searches for shared libraries at run time

5. Executable and Linking Format (ELF)

- ELF sections
- ELF as a template for construction of a process
- *objdump* and *readelf*
- Loading a program at run time
- Role of the dynamic linker

6. Dynamically loaded libraries

- The *dlopen* API (*dlopen()*, *dlsym()*, *dladdr()*, *dlsym()*, and more)

7. Controlling symbol visibility

- Why controlling symbol visibility matters
- `static` and `visibility hidden`
- Using version scripts to control symbol visibility
- `gcc` options for controlling symbol visibility

8. Symbol versioning

- Symbol versioning using version scripts
- Advantages of symbol versioning

9. Preloading shared libraries

- Using `LD_PRELOAD` to load shared libraries
- Uses of preloading