

Installation of Elmer

Obtaining the source, compiling, installing and setting up the system

Thomas Zwinger

`thomas.zwinger[at]csc.fi`

Computational Environment & Application

CSC—Scientific Computing Ltd.

The Finnish IT center for science

Espoo, Finland



Contents

The Source of Elmer

Binary on Ubuntu/Linux

Binary on Win32

FUNET FTP

SourceForge SVN

Compilation of Elmer

UNIX/Linux

UNIX/Linux MPI

Win32 using MinGw and g95

Elmer Web



The Source of Elmer

- Elmer is available either as **binary**:
Win32 ([SourceForge](#) or [Funet](#)[†]), Ubuntu/Linux ([Funet](#))

[†] Funet is maintained by CSC, hence a CSC-internal source

The Source of Elmer

- Elmer is available either as **binary**:
Win32 ([SourceForge](#) or [Funet](#)[†]), Ubuntu/Linux ([Funet](#))
- + easy-go solution

[†] Funet is maintained by CSC, hence a CSC-internal source

The Source of Elmer

- Elmer is available either as **binary**:
 - Win32 ([SourceForge](#) or [Funet[†]](#)), Ubuntu/Linux ([Funet](#))
- + easy-go solution
- not easy to update, no source - no hack!

[†] Funet is maintained by CSC, hence a CSC-internal source

The Source of Elmer

- Elmer is available either as **binary**:
 - Win32 ([SourceForge](#) or [Funet[†]](#)), Ubuntu/Linux ([Funet](#))
 - + easy-go solution
 - not easy to update, no source - no hack!
- or as **source** (~ 300k lines of code) from either Funet, SourceForge as [download](#) or using subversion

```
svn co https://elmerfem.svn.sourceforge.net/svnroot/elmerfem
elmerfem
```

[†] Funet is maintained by CSC, hence a CSC-internal source

The Source of Elmer

- Elmer is available either as **binary**:
 - Win32 ([SourceForge](#) or [Funet[†]](#)), Ubuntu/Linux ([Funet](#))
 - + easy-go solution
 - not easy to update, no source - no hack!
- or as **source** (~ 300k lines of code) from either Funet, SourceForge as [download](#) or using subversion

```
svn co https://elmerfem.svn.sourceforge.net/svnroot/elmerfem
elmerfem
```

- compiling needs patience and endurance

[†] Funet is maintained by CSC, hence a CSC-internal source

The Source of Elmer

- Elmer is available either as **binary**:
Win32 ([SourceForge](#) or [Funet](#)[†]), Ubuntu/Linux ([Funet](#))
 - + easy-go solution
 - not easy to update, no source - no hack!
- or as **source** (~ 300k lines of code) from either Funet, SourceForge as [download](#) or using subversion

```
svn co https://elmerfem.svn.sourceforge.net/svnroot/elmerfem
elmerfem
```

- compiling needs patience and endurance
- + easy update, code is best documentation, customisation

[†] Funet is maintained by CSC, hence a CSC-internal source

Binary on Ubuntu/Linux

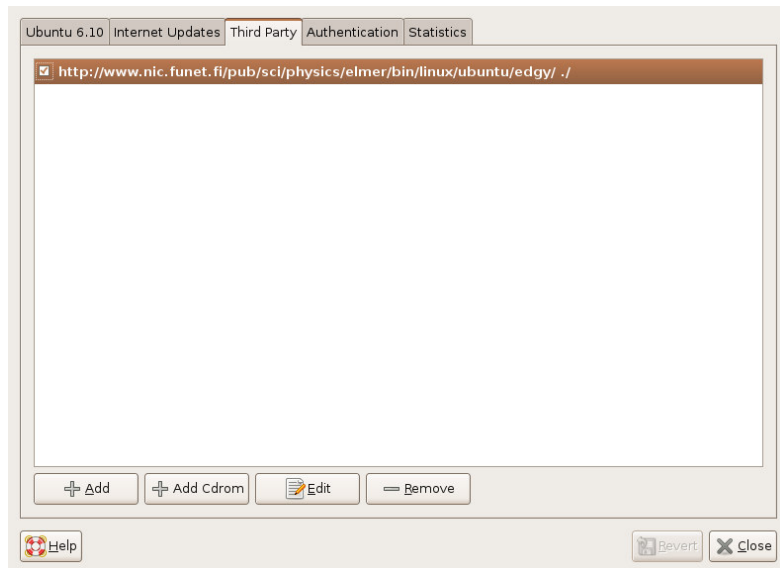
In **Synaptic Package Manager**:

- open menu **Settings** → **Repositories**

Binary on Ubuntu/Linux

In Synaptic Package Manager:

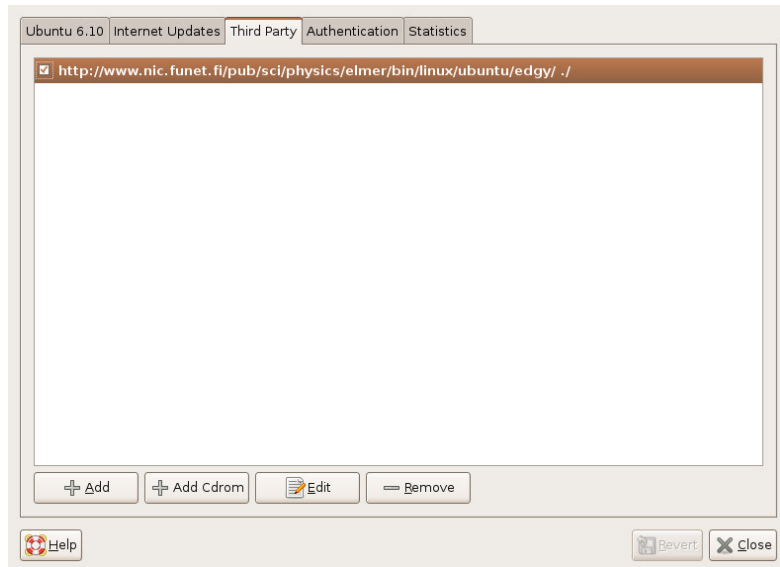
- open menu **Settings** → **Repositories**
- choose **+Add** and add the Funet-server



Binary on Ubuntu/Linux

In Synaptic Package Manager:

- open menu **Settings** → **Repositories**
- choose **+Add** and add the Funet-server



- then **Search** for keyword *elmer* and toggle to add

Binary on Win32

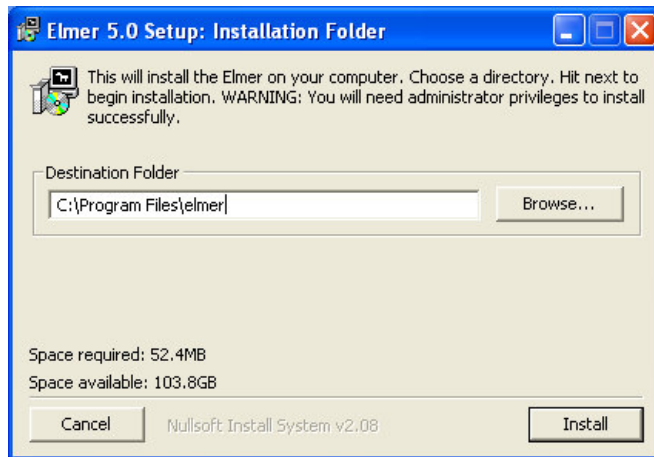
<http://www.csc.fi/english/pages/elmer/download/quickstart/index.htm>

- download the installer from [SourceForge](#)

Binary on Win32

http://www.csc.fi/english/pages/elmer/download/quickstart/index_htm

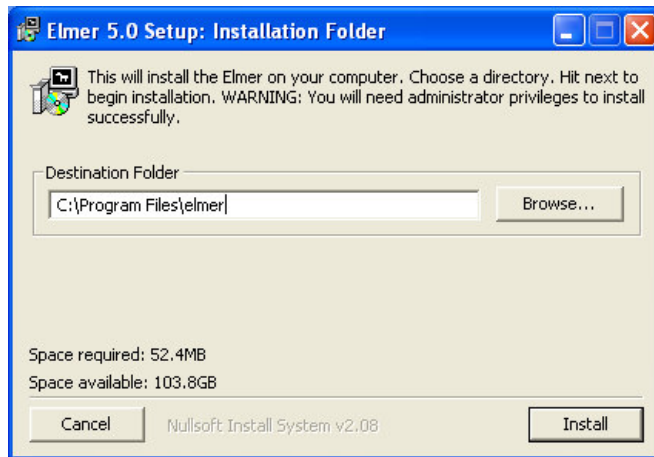
- download the installer from [SourceForge](#)
- double-click the installer - define directory:



Binary on Win32

http://www.csc.fi/english/pages/elmer/download/quickstart/index_htm

- download the installer from **SourceForge**
- double-click the installer - define directory:



- executables are then called from **Start**→**All Programs**→**Accessories**→**Command prompt**

What Binaries?

ElmerSolver FEM Solver with all the nice features built in (parallel version ElmerSolver_mpi). The different applications (like Navier-Stokes, Heat Transfer, ...) are dynamically linked libraries (.dll, .so)

What Binaries?

- ElmerSolver** FEM Solver with all the nice features built in (parallel version ElmerSolver_mpi). The different applications (like Navier-Stokes, Heat Transfer, ...) are dynamically linked libraries (.dll, .so)
- ElmerPost** Versatile and self-explaining postprocessor based on Mesa and TCL/TK graphic libraries

What Binaries?

- ElmerSolver** FEM Solver with all the nice features built in (parallel version ElmerSolver_mpi). The different applications (like Navier-Stokes, Heat Transfer, ...) are dynamically linked libraries (.dll, .so)
- ElmerPost** Versatile and self-explaining postprocessor based on Mesa and TCL/TK graphic libraries
- ElmerFront** Graphical user interface for creating setups for simple problems

What Binaries?

- ElmerSolver** FEM Solver with all the nice features built in (parallel version ElmerSolver_mpi). The different applications (like Navier-Stokes, Heat Transfer, ...) are dynamically linked libraries (.dll, .so)
- ElmerPost** Versatile and self-explaining postprocessor based on Mesa and TCL/TK graphic libraries
- ElmerFront** Graphical user interface for creating setups for simple problems
- ElmerGrid** Simple mesh generator and import-export filter

What Binaries?

- ElmerSolver** FEM Solver with all the nice features built in (parallel version ElmerSolver_mpi). The different applications (like Navier-Stokes, Heat Transfer, ...) are dynamically linked libraries (.dll, .so)
- ElmerPost** Versatile and self-explaining postprocessor based on Mesa and TCL/TK graphic libraries
- ElmerFront** Graphical user interface for creating setups for simple problems
- ElmerGrid** Simple mesh generator and import-export filter
- Mesh2D** Delaunay (2D) mesh generator called by Elmer-Front

What Binaries?

ElmerSolver	FEM Solver with all the nice features built in (parallel version ElmerSolver_mpi). The different applications (like Navier-Stokes, Heat Transfer, ...) are dynamically linked libraries (.dll, .so)
ElmerPost	Versatile and self-explaining postprocessor based on Mesa and TCL/TK graphic libraries
ElmerFront	Graphical user interface for creating setups for simple problems
ElmerGrid	Simple mesh generator and import-export filter
Mesh2D	Delaunay (2D) mesh generator called by ElmerFront
ViewFactors	computation of view factors

Funet FTP

<http://www.nic.funet.fi/pub/sci/physics/elmer/>

<ftp://ftp.nic.funet.fi/pub/sci/physics/elmer/>

Index of /pub/sci/physics/elmer

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 bin/	15-May-2007 10:15	-	
 doc/	09-Jan-2008 16:43	-	
 src/	28-Sep-2007 14:30	-	
 README	13-Feb-2007 12:31	865	

SourceForge SVN

<http://sourceforge.net/projects/elmerfem/>

The screenshot shows the SourceForge website interface. At the top, the 'SOURCEFORGE.NET' logo is on the left, and 'Log in', 'Create account', and 'Help' links are on the right. A navigation bar contains 'Home', 'Browse Software', 'Marketplace', 'Community', 'Create Project', and 'Jobs'. Below this is a search bar with a dropdown menu set to 'Software' and buttons for 'Search' and 'Advanced'. The main content area features several advertisements: 'CFdesign - Upfront CFD' (Fluid flow and thermal simulations for mechanical engineers, www.valeng.com), 'Finite Element Analysis' (Design & Fitness-For-Service Finite Element Analysis, www.smpes.com), and 'Gantt Chart Components' (ActiveX, .NET and Java Controls for interactive Gantt applications, www.netronic.com). Below the ads, the breadcrumb 'SF.net » Projects » Elmer-fem » Summary' is shown. The project title 'Elmer-fem' is prominently displayed, followed by a navigation bar with 'Project', 'Tracker', 'Code', 'Services', 'Download', and 'Wiki' tabs. To the right of the project title are links for 'Project Web Site', 'Stats', and 'RSS'. A paragraph describes Elmer as a collection of finite element programs, libraries, and visualization tools for numerical solutions of partial differential equations and multiphysical problems, with a link to the main page at http://www.csc.fi/elmer. Below this is an advertisement for HP with the text '- Enter Here to Research Featured Solutions -'. Further down, another 'Ads by Google' section contains three ads: 'MSc Computational Maths' (Challenging course in modern numerical analysis, brnel.ac.uk/mathspg), 'Free 3D Software Download' (3D animation & modeling software, Easy to use. Not shareware, www.freerissoftware.com), and 'Fluid Dynamics Software' (New, easy, low-cost CFD software Embedded in your MCAD environment, www.flomerics.com/products/EFD). At the bottom left, there is a 'Download Elmer-fem' button and a 'Get the most from Open Source' button. The 'Download Elmer-fem' section lists project admins (apursula, juhar, juhavierinen, mlsf, mmalinen, raback, tzwinger), operating system (None Listed), license (GNU General Public License (GPL)), and category (Mathematics, Physics, Visualization). The 'Get the most from Open Source' section promotes expert services from Sourceforge.net Marketplace, supported by the community. The SourceForge.NET Marketplace logo is also visible.



Which packages?

fem	source for the essential parts of ElmerSolver (F90)
elmerpost	source for ElmerPost
elmergrid	source for ElmerGrid
front	source for ElmerFront
meshgen2d	source code of the 2D Delaunay mesher Mesh2d

Which packages?

fem	source for the essential parts of ElmerSolver (F90)
elmerpost	source for ElmerPost
elmergrid	source for ElmerGrid
front	source for ElmerFront
meshgen2d	source code of the 2D Delaunay mesher Mesh2d
eio	source for Elmer input/output library written in C++
mathlibs	contains basic mathematical libraries such as Lapack, Blas, Arpack, and Parpack
hutiter	source for iterative linear algebra solvers called by ElmerSolver (mostly F90)
matc	source for command file interpreter of ElmerSolver and inside the command window of ElmerPost (C)
umfpack	source code of the Umfpack library from University of California

Which packages?

fem	source for the essential parts of ElmerSolver (F90)
elmerpost	source for ElmerPost
elmergrid	source for ElmerGrid
front	source for ElmerFront
meshgen2d	source code of the 2D Delaunay mesher Mesh2d
eio	source for Elmer input/output library written in C ⁺⁺
mathlibs	contains basic mathematical libraries such as Lapack, Blas, Arpack, and Parpack
hutiter	source for iterative linear algebra solvers called by ElmerSolver (mostly F90)
matc	source for command file interpreter of ElmerSolver and inside the command window of ElmerPost (C)
umfpack	source code of the Umfpack library from University of California
doc	the \LaTeX -based documentation input files
elmer-doc	help documents of Elmer

Compilation of Elmer

- each package contains its own **configure** script

Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir



Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir
- thereafter: `make` and `make install`



Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir
- thereafter: `make` and `make install`
- Prerequisites:

Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir
- thereafter: `make` and `make install`
- Prerequisites:
 1. A fair amount of patience



Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir
- thereafter: `make` and `make install`
- Prerequisites:
 1. A fair amount of patience
 2. Fortran 90, C and C⁺⁺ compiler

Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir
- thereafter: `make` and `make install`
- Prerequisites:
 1. A fair amount of patience
 2. Fortran 90, C and C⁺⁺ compiler
 3. autoconf, automake

Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir
- thereafter: `make` and `make install`
- Prerequisites:
 1. A fair amount of patience
 2. Fortran 90, C and C⁺⁺ compiler
 3. autoconf, automake
- Optional:

Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir
- thereafter: `make` and `make install`
- Prerequisites:
 1. A fair amount of patience
 2. Fortran 90, C and C⁺⁺ compiler
 3. autoconf, automake
- Optional:
 1. for parallel version MPI library (like OpenMPI) and Hypre

Compilation of Elmer

- each package contains its own **configure** script
- usually it is enough to just say: `./configure --prefix=$ELMER_HOME`
in each sub-dir
- thereafter: `make` and `make install`
- Prerequisites:
 1. A fair amount of patience
 2. Fortran 90, C and C⁺⁺ compiler
 3. autoconf, automake
- Optional:
 1. for parallel version MPI library (like OpenMPI) and Hypre
 2. `ffmpeg` and `libjpeg` to use the MPEG/JPEG plug-in in ElmerPost

UNIX/Linux

```
#!/bin/sh -f
#the compiler (here the gcc 4.X suite)
export CC=gcc
export CXX=g++
export FC=gfortran
export F77=gfortran
#the compiler flags
export CFLAGS=" "
export CXXFLAGS=" "
export FCFLAGS=" "
export F77FLAGS=" "
export FFLAGS=" "
#linking
export LDFLAGS=" "
#paths
export ELMER_HOME="/path/to/Elmerdir"
```



UNIX/Linux contd.

```
# make the modules
modules="matc umfpack mathlibs elmergrid meshgen2d eio hutiter
fem post"
# configure and build
for m in $modules; do
cd $m ; ./configure --prefix=$ELMER_HOME && make clean && make
&& make install && cd ..
done
```



UNIX/Linux contd.

```
# make the modules
modules="matc umfpack mathlibs elmergrid meshgen2d eio hutiter
fem post"
# configure and build
for m in $modules; do
cd $m ; ./configure --prefix=$ELMER_HOME && make clean && make
&& make install && cd ..
done
```

● Additional arguments for configure:

- debugging build: `--with-debug`
- 32-bit platform: `-with-64bits=no`

UNIX/Linux contd.

```
# make the modules
modules="matc umfpack mathlibs elmergrid meshgen2d eio hutiter
fem post"
# configure and build
for m in $modules; do
cd $m ; ./configure --prefix=$ELMER_HOME && make clean && make
&& make install && cd ..
done
```

- Additional arguments for configure:
 - debugging build: `--with-debug`
 - 32-bit platform: `-with-64bits=no`
- adapt the paths in the script above to match your system!
- save it into `compile.sh` in the root of the source-tree
- run the compilation `./compile.sh`

UNIX/Linux parallel MPI

- compile and install a MPI library; e.g. [OpenMPI](#)

UNIX/Linux parallel MPI

- compile and install a MPI library; e.g. [OpenMPI](#)
- Optionally, [Hypre](#) can be compiled and installed

UNIX/Linux parallel MPI

- compile and install a MPI library; e.g. **OpenMPI**
- Optionally, **Hypre** can be compiled and installed

```
#!/bin/sh -f
#the OpenMPI wrapper
export CC=mpicc
export CXX=mpic++
export FC=mpif90
export F77=mpif90
#the compiler flags
export CFLAGS="-I/usr/local/hypre2.0/include"
export FCFLAGS="-I/usr/local/hypre2.0/include"
export F77FLAGS="-I/usr/local/hypre2.0/include"
export FFLAGS="-I/usr/local/hypre2.0/include"
#linking needs Hypre included
export LDFLAGS="L/usr/local/hypre2.0/lib -lHYPRE"
```

UNIX/Linux parallel MPI contd.

```
#paths; linking needs Hyper and MPI
export ELMER_HOME="/path/to/Elmerdir"
export LD_LIBRARY_PATH="/usr/local/OpenMPI/lib:$LD_LIBRARY_PATH"
export LD_LIBRARY_PATH="/usr/local/hypre2.0/lib:$LD_LIBRARY_PATH"
# make the modules
modules="matc umfpack mathlibs elmergrid meshgen2d eio hutiter
fem post"
# configure and build
for m in $modules; do
cd $m ; ./configure --prefix=$ELMER_HOME
--with-mpi-dir=$MPI_HOME && make clean && make && make install
&& cd ..
done
```



Win32 using MinGw and g95[†]

- Get the installer MinGW-5.1.3.exe from [SourceForge](#) and run it
Install the "current" version
Choose all possible packages to download and install
Install in C:\MinGW

[†] Contribution by Mikko Lyly

Win32 using MinGw and g95[†]

- Get the installer MinGW-5.1.3.exe from [SourceForge](#) and run it
Install the "current" version
Choose all possible packages to download and install
Install in C : \MinGW
- Get the installer MSYS-1.0.10.exe and run it
Install in C : \msys\1.0

[†] Contribution by Mikko Lyly

Win32 using MinGw and g95[†]

- Get the installer MinGW-5.1.3.exe from [SourceForge](#) and run it
Install the "current" version
Choose all possible packages to download and install
Install in C:\MinGW
- Get the installer MSYS-1.0.10.exe and run it
Install in C:\msys\1.0
- Get the installer msysDTK-1.0.1.exe and run it

[†] Contribution by Mikko Lyly

Win32 using MinGw and g95[†]

- Get the installer MinGW-5.1.3.exe from [SourceForge](#) and run it
Install the "current" version
Choose all possible packages to download and install
Install in C:\MinGW
- Get the installer MSYS-1.0.10.exe and run it
Install in C:\msys\1.0
- Get the installer msysDTK-1.0.1.exe and run it
- Download the GCC 4 packages of MinGW
the contents of these packages in /mingw:
`gcc-core-4.2.1-sjlj-2.tar.gz`
`gcc-g++-4.2.1-sjlj-2.tar.gz`
`gcc-gfortran-4.2.1-sjlj-2.tar.gz`

[†] Contribution by Mikko Lyly

Win32 using MinGw and g95 contd.

- Download and run the installer [g95-MinGW.exe](#) for g95
(mind: g95 \neq gfortran)

Win32 using MinGw and g95 contd.

- Download and run the installer [g95-MinGW.exe](#) for g95
(mind: g95 \neq gfortran)
- Let the installer extract its content into the MinGW structure

Win32 using MinGw and g95 contd.

- Download and run the installer [g95-MinGW.exe](#) for g95 (mind: g95 \neq gfortran)
- Let the installer extract its content into the MinGW structure
- test your compiler:

```
$ cat > test.c
/* test.c */
#include <stdio.h>
int main() {
    fprintf(stdout, "It works!");
}
<CTRL-D>
$ gcc-sjlj -O -o test.exe test.c
$ ./test.exe
It works!
:
:
```

Win32 using MinGw and g95 contd.

- Download and run the installer [g95-MinGW.exe](#) for g95 (mind: g95 \neq gfortran)
- Let the installer extract its content into the MinGW structure
- test your compiler:

```
$ cat > test.c
/* test.c */
#include <stdio.h>
int main() {
    fprintf(stdout, "It works!");
}
<CTRL-D>
$ gcc-sjlj -O -o test.exe test.c
$ ./test.exe
It works!
:
:
```

- also test your fortran compiler!



Win32 using MinGw and g95 contd.

```
#!/bin/sh
export ELMER_HOME=/c/Elmer5.4
export CC=gcc
export CXX=g++
export FC=g95
export F77=g95
export LDFLAGS="-L/mingw/lib/gcc/mingw32/3.4.5/ -Xlinker
--stack=1000000000 "
modules="matc umfpack mathlibs elmergrid meshgen2d eio hutiter
fem"
for m in $modules; do
cd $m
./configure --prefix=$ELMER_HOME
make
make install
cd ..
done
```



Elmer Web



The screenshot shows a web browser window with the address bar containing `http://www.csc.fi/elmer`. The browser's menu bar includes File, Edit, View, Go, Bookmarks, Tools, and Help. Below the address bar, there are navigation icons and a list of folders: Support, Shop, Products, Training, CSC, Bildung/Wissenschaft, Bibliothek, Services, and Computer. The page content features the Elmer logo (a colorful circular icon) and the text "Elmer" in a large font. To the right, there is a link for "CSC's homepage" and the CSC logo. A horizontal separator line is present. Below it, the word "Elmer" is displayed. On the left side, there is a vertical navigation menu with the following items: Binaries, Documentation, Sources and compilation, Application examples, Interfaces, White papers, Mailing lists, Links, News, and Services and contact. The main content area has the heading "Elmer" followed by the sub-heading "Open Source Finite Element Software for Multiphysical Problems". A link "Tämä sivu suomeksi" is visible. The text describes Elmer as an open source multiphysical simulation software developed by CSC, starting in 1995 in collaboration with Finnish Universities, research institutes, and industry. It lists various physical models: fluid dynamics, structural mechanics, electromagnetics, heat transfer, and acoustics. A 3D visualization of a melt flow in a Czochralski growth of silicon is shown, with a caption: "Temperature distribution of melt flow in Czochralski growth of silicon." The footer contains the copyright notice "© CSC" and the date "Last modified 5.2.2008". The browser's status bar at the bottom shows "Done".

File Edit View Go Bookmarks Tools Help

http://www.csc.fi/elmer

Support Shop Products Training CSC Bildung/Wissenschaft Bibliothek Services Computer

Suomeksi CSC's homepage

 Elmer 

Elmer

Binaries
Documentation
Sources and compilation
Application examples
Interfaces
White papers
Mailing lists
Links
News
Services and contact

Elmer

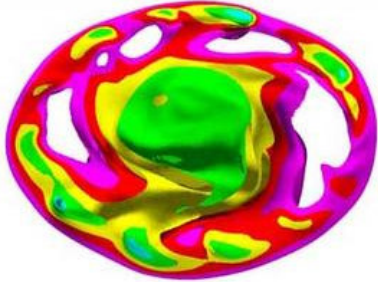
Tämä sivu suomeksi

Open Source Finite Element Software
for Multiphysical Problems

Elmer is an open source multiphysical simulation software developed by CSC. Elmer development was started 1995 in collaboration with Finnish Universities, research institutes and industry.

Elmer includes physical models of fluid dynamics, structural mechanics, electromagnetics, heat transfer and acoustics, for example. These are described by partial differential equations which Elmer solves by the Finite Element Method (FEM).

These pages are intended to give information on the Elmer software and to improve the information transfer in the Elmer community.



Temperature distribution of melt flow in Czochralski growth of silicon.

© CSC Last modified 5.2.2008

Done