Laurent Deniau, CERN BE/ABP, 1211 Geneva 23, laurent.deniau@cern.ch

Beams Department

Methodical Accelerator Design

Project Status Report

Laurent Deniau

CERN-BE/ABP

29th November 2011

Achievements

- Files split continued, still 66 modules (`madx?.c` and `mad_gfun.h` removed!)
- Few bugs identified and corrected (classification will come soon)
- New build process (10 compilers, 4 OS, 32/64 bit, parallel compilation)
- C `main()` now fully supported for all compilers
- All wrappers removed (fragile), Fortran <=> C interface cleaned
  - `mad_wrap_f.h` removed, `mad_extrn_f.h` added
- 1000+ warnings corrected
- 6 modules functionally closed (6track, sdds, sodd, rplot, mkthin, sxf)
- X11/no-X11 better managed (`gxx11c.c` vs `gxx11psc.c`)
- VERSION file added, automatic build of madx strings
- Source code moved into subdirectory `madX/src` (files history kept)
- 1 development release 5.00.09 (first with new file structure)
- Bug tracker under cleaning and sorting (to be moved on svntrack)
- Mailing lists filled
- Web sites under redesign

E-groups

- Populated last Friday
- `mad-dev@cern.ch`
  - **Developers e-group** (static)
  - 8 members: 2 ABP-LCU + 5 not-LCU
- `mad-usr@cern.ch`
  - **Users e-group** (semi-static)
  - Includes be-dep-abp-lcu@cern.ch (33 members, dynamic)
  - Includes `mad-dev@cern.ch`
  - 72 members: 33 ABP-LCU + 32 not-LCU + 7 not-CERN
- `mad-pub@cern.ch`
  - **Public e-group** (semi-static)
  - Includes be-dep-abp@cern.ch (198 members, dynamic)
  - Includes `mad-usr@cern.ch`
  - 223 members: 198 ABP + 18 not-ABP + 7 not-CERN

Build System 1/2

- Fully customizable, parallel compilation, easy to configure and used (see README files)
  - ➡ `make -j5 ARCH=64 ONLINE=yes STATIC=yes CC=icc CXX=icc FC=ifort`
- **10 compilers** (12 orthogonal configurations, can be intermixed, support X-compilation)
  - ➡ C: gcc, icc, icl, cl
  - ➡ C++: g++, icc, icl, cl
  - ➡ F90: gfortran, ifort, lf95-32, lf95-64, nagfor, g95
- **4 platforms** (same build system!)
  - ➡ Linux, Mac OS X, Windows, Cygwin32
- **2 architectures**
  - ➡ 32 / 64 bit
- Based on GNU make (Unix), equipped for debugging (`SHOW=yes`)
  - ➡ Windows: http://gnuwin32.sourceforge.net/packages/make.htm

Unified compilation
+
Unified releases

Release Config

Makefile

Project Config

Makefile_cpp
Makefile_c
Makefile_cxx
Makefile_f90
Makefile_lib
Makefile_sys
Makefile_pre
Makefile_post

Build System Files

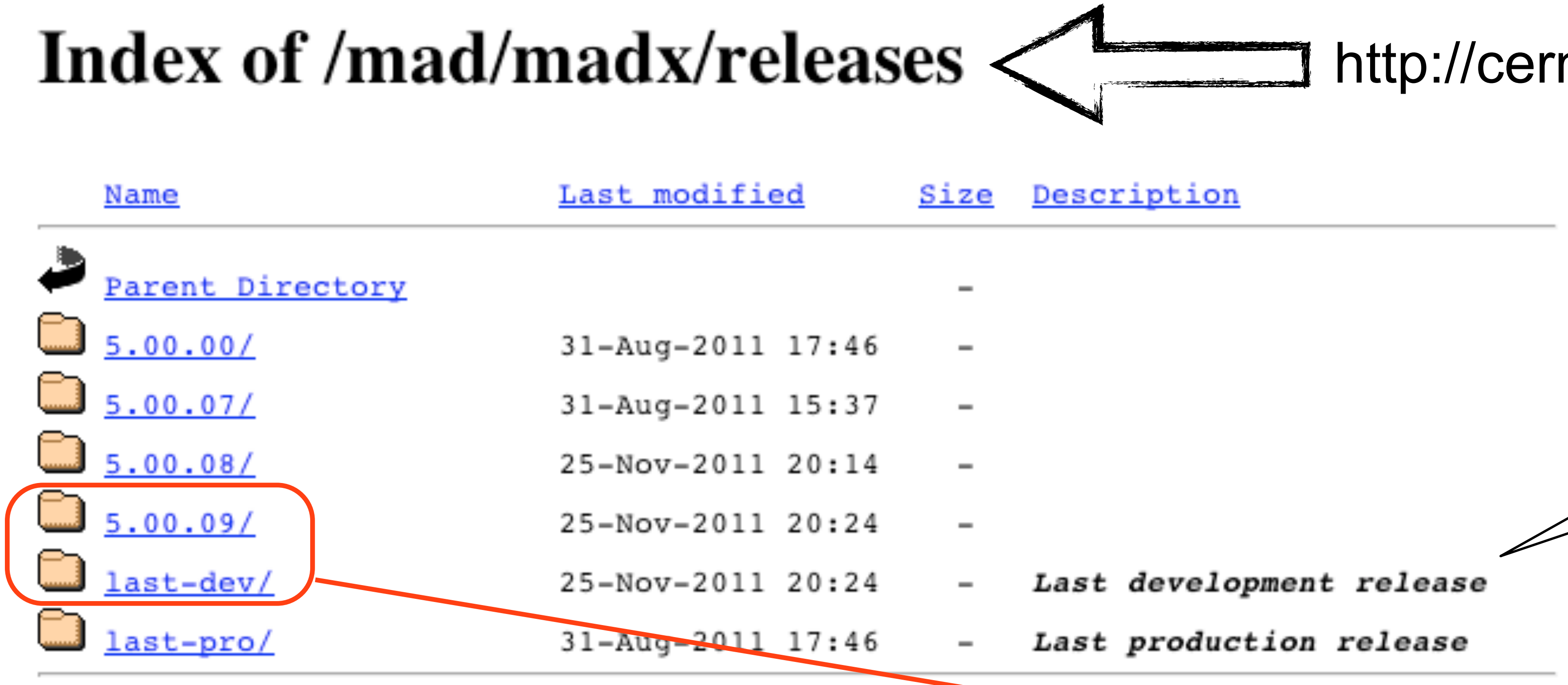
README
README.lxplus
README.windows
clean.rules
compiler.cl
compiler.g++
compiler.g95
compiler.gcc
compiler.gfortran
compiler.icc
compiler.icl
compiler.ifort
compiler.lf95
compiler.nagfor
compiler.rules
depend.rules
info.rules
linker.cl
linker.g++
linker.g95
linker.gcc
linker.gfortran
linker.icc
linker.icl
linker.ifort
linker.ld
linker.lf95
linker.nagfor
linker.rules
make.cfg
make.inc
make.lib
system.Cygwin
system.Darwin
system.Linux
system.Windows

Build System 2/2

```
#ifdef _GFORTRAN
void _gfortran_set_args    (int, char *[]);
void _gfortran_set_options (int, int   []);
#endif

#ifdef _NAGFOR
void f90_init   (int, char *[]);
void f90_finish (int);
#endif

#ifdef _G95
void g95_runtime_start (int, char *[]);
void g95_runtime_stop  (void);
#endif

#ifdef _LF95
// Lahey f95 specific (requires main to be MAIN__)
int MAIN__()
{
#else
int main(int argc, char *argv[])
{
  mad_stck_base = &argc;
  mad_argc = argc;
  mad_argv = argv;
#endif
```

```
#ifdef _GFORTRAN
  _gfortran_set_args(argc, argv);
  _gfortran_set_options(0, 0);
#endif

#ifdef _NAGFOR
  f90_init(argc, argv);
#endif

#ifdef _G95
  g95_runtime_start(argc, argv);
#endif

// madx main program
  madx_start();
  madx_input(CALL_LEVEL_ZERO);
  madx_finish();

#ifdef _NAGFOR
  f90_finish(EXIT_SUCCESS);
#endif

#ifdef _G95
  g95_runtime_stop();
#endif

  return EXIT_SUCCESS;
}
```

Releases / Downloads

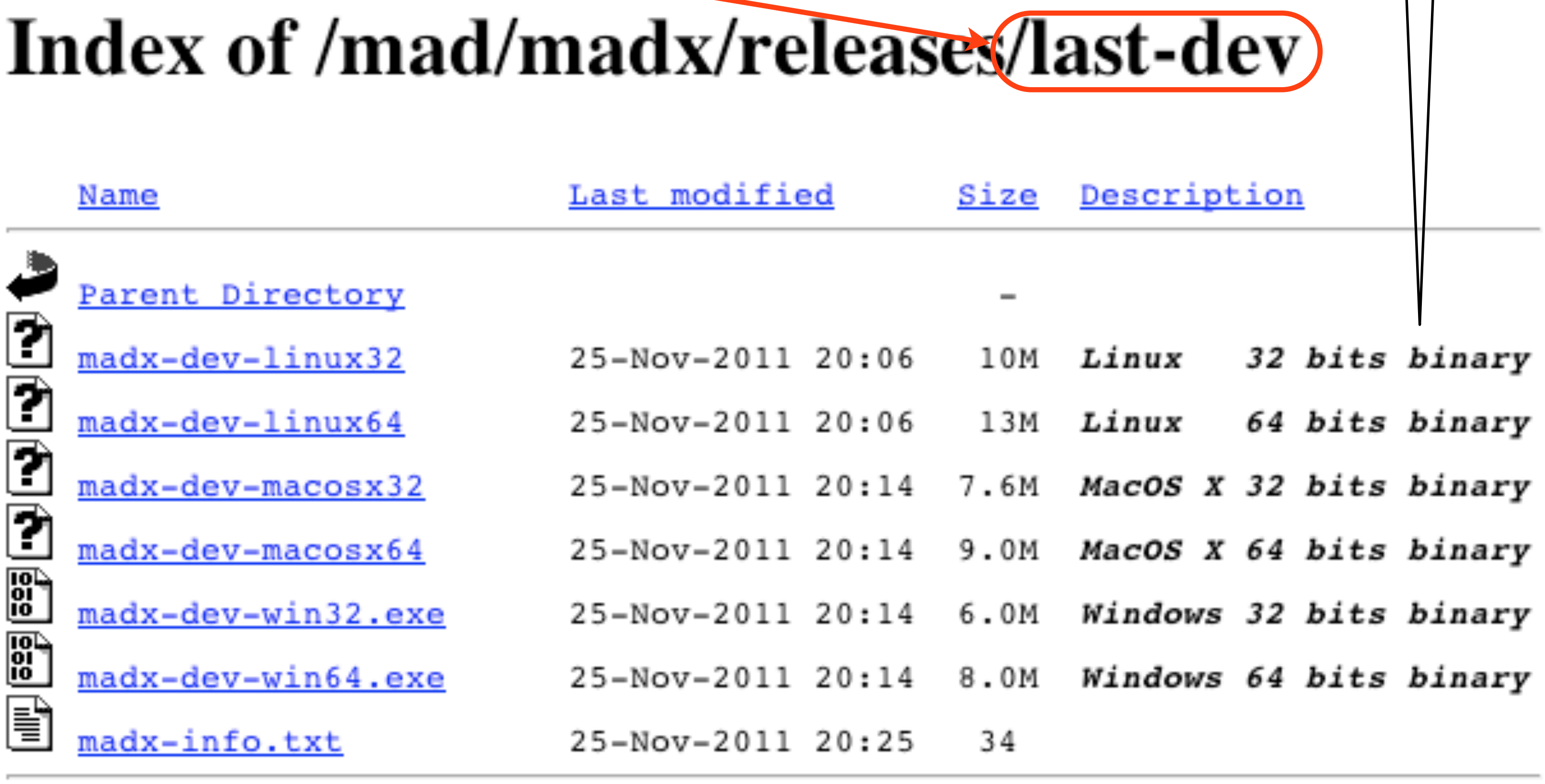


Index of /mad/madx/releases

Name	Last modified	Size	Description
Parent Directory		-	
5.00.00/	31-Aug-2011 17:46	-	
5.00.07/	31-Aug-2011 15:37	-	
5.00.08/	25-Nov-2011 20:14	-	
5.00.09/	25-Nov-2011 20:24	-	
last-dev/	25-Nov-2011 20:24	-	*Last development release*
last-pro/	31-Aug-2011 17:46	-	*Last production release*

http://cern.ch/mad/madx/releases

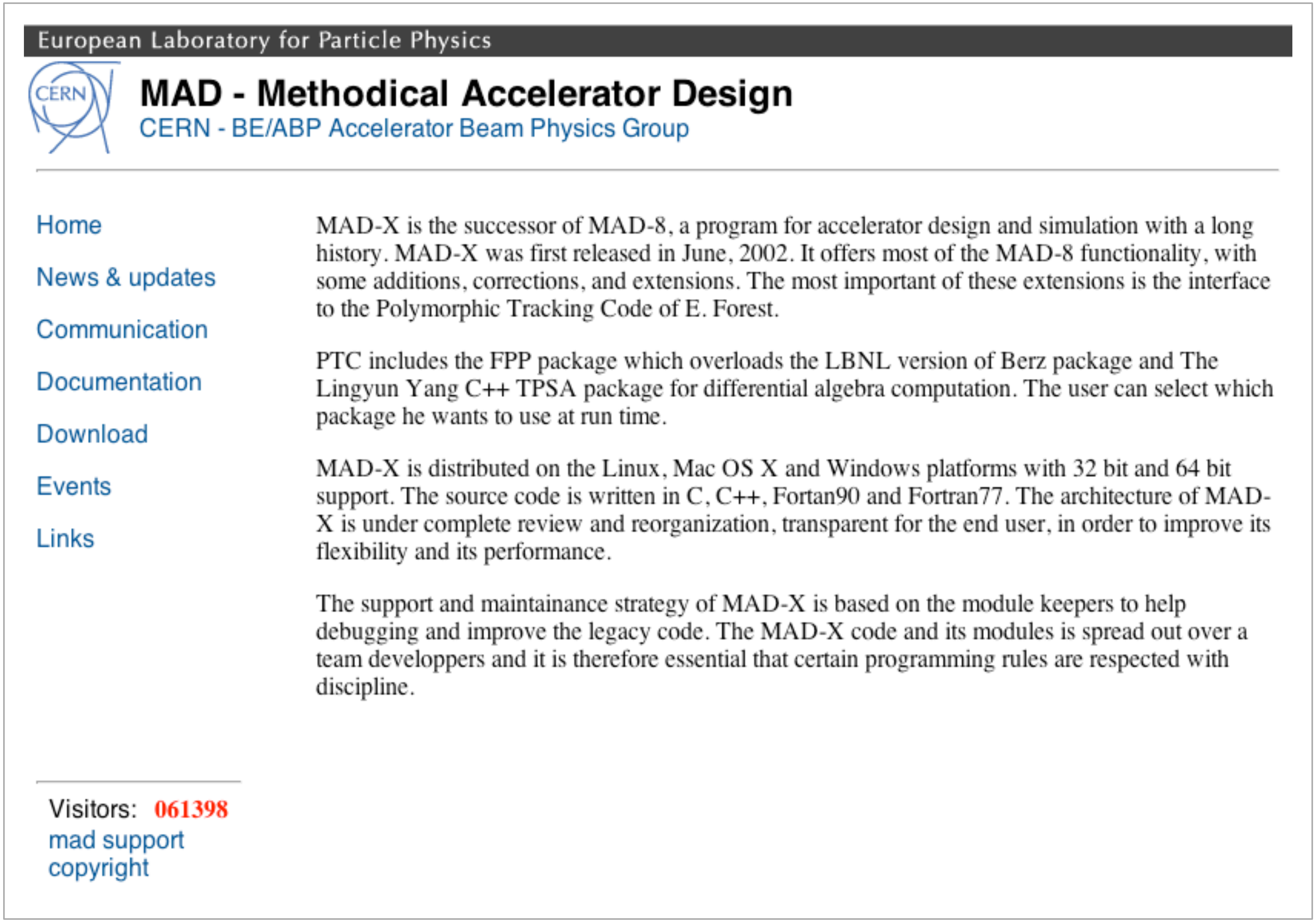
Release types



Index of /mad/madx/releases/last-dev

Name	Last modified	Size	Description
Parent Directory		-	
madx-dev-linux32	25-Nov-2011 20:06	10M	*Linux 32 bits binary*
madx-dev-linux64	25-Nov-2011 20:06	13M	*Linux 64 bits binary*
madx-dev-macosx32	25-Nov-2011 20:14	7.6M	*MacOS X 32 bits binary*
madx-dev-macosx64	25-Nov-2011 20:14	9.0M	*MacOS X 64 bits binary*
madx-dev-win32.exe	25-Nov-2011 20:14	6.0M	*Windows 32 bits binary*
madx-dev-win64.exe	25-Nov-2011 20:14	8.0M	*Windows 64 bits binary*
madx-info.txt	25-Nov-2011 20:25	34	

Web Site



European Laboratory for Particle Physics

MAD - Methodical Accelerator Design

CERN - BE/ABP Accelerator Beam Physics Group

- Home
- News & updates
- Communication
- Documentation
- Download
- Events
- Links

MAD-X is the successor of MAD-8, a program for accelerator design and simulation with a long history. MAD-X was first released in June, 2002. It offers most of the MAD-8 functionality, with some additions, corrections, and extensions. The most important of these extensions is the interface to the Polymorphic Tracking Code of E. Forest.

PTC includes the FPP package which overloads the LBNL version of Berz package and The Lingyun Yang C++ TPSA package for differential algebra computation. The user can select which package he wants to use at run time.

MAD-X is distributed on the Linux, Mac OS X and Windows platforms with 32 bit and 64 bit support. The source code is written in C, C++, Fortan90 and Fortran77. The architecture of MAD-X is under complete review and reorganization, transparent for the end user, in order to improve its flexibility and its performance.

The support and maintainance strategy of MAD-X is based on the module keepers to help debugging and improve the legacy code. The MAD-X code and its modules is spread out over a team developpers and it is therefore essential that certain programming rules are respected with discipline.

Visitors: 061398
mad support
copyright

Next Steps

- Finish the web site (feedback will be welcome)
- Add source code tarball to releases directory
- Move documentation into SVN (synchronize release vs doc)
- Move bugs from JIRA to SvnTrack (keep history)
- Classify and document the bugs (ongoing)
- Close more modules (background task)
- Extend the build system with a test system (urgent)

- Prepare madx for crabcav and thick quadrupoles (track, twiss, mkthin, 6track)
  - ➡ A. Latina (20%)